Effective Pairings

in Isogeny-based Cryptography

Krijn Reijnders

LATINCRYPT 2023

Radboud University

Pairings map
elliptic curve problems
to finite field problems

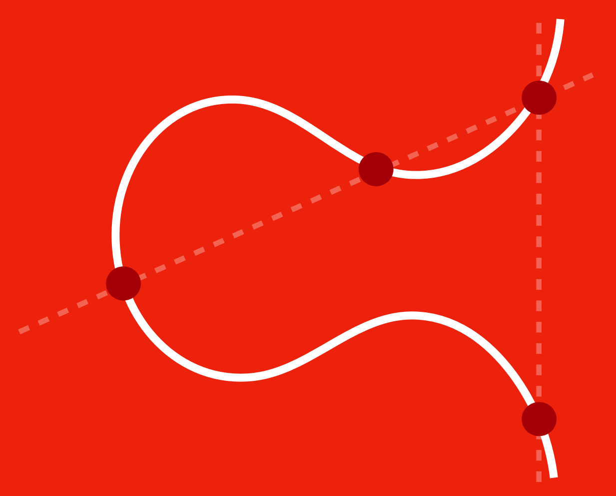
Elliptic curve arithmetic
is slow

Finite field arithmetic
is (very) fast

Hence, fast pairings
means fast solutions

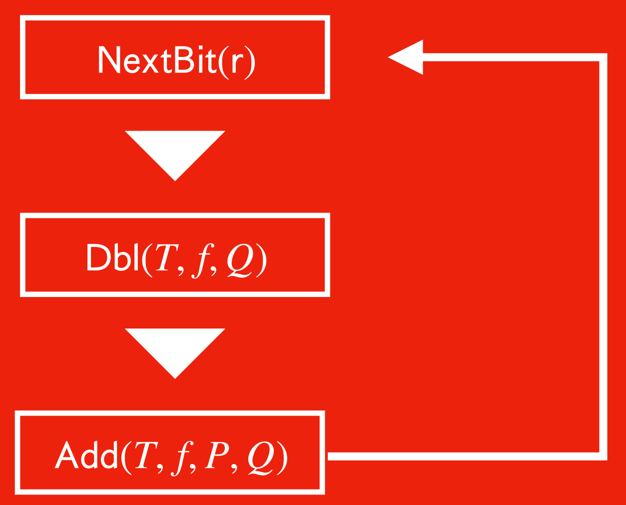
Effective Pairings in Isogeny-based Cryptography

1



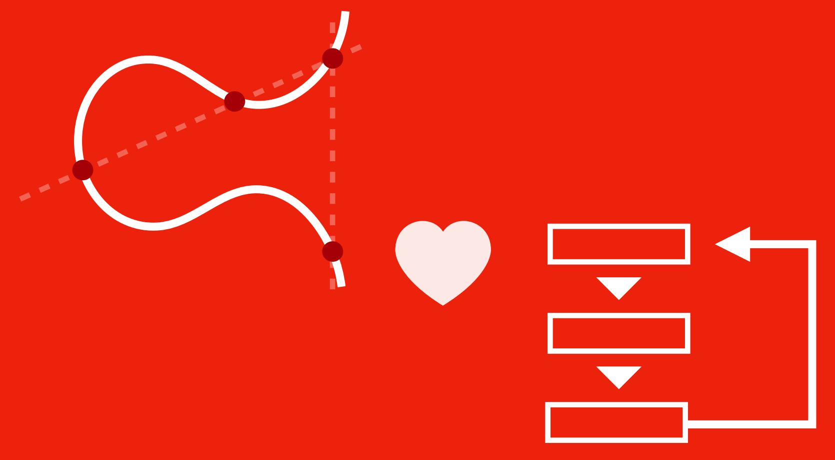
Isogenies & Pairings

2



Speeding-up general pairings

3



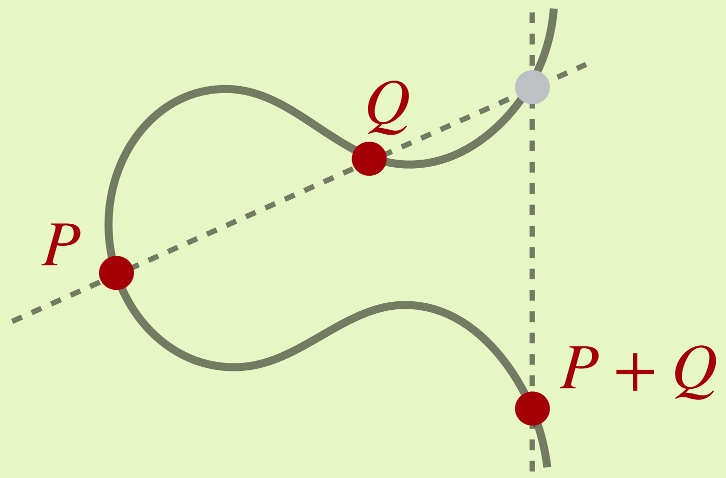
Applying pairings in isogeny crypto

What are pairings and what are isogenies?

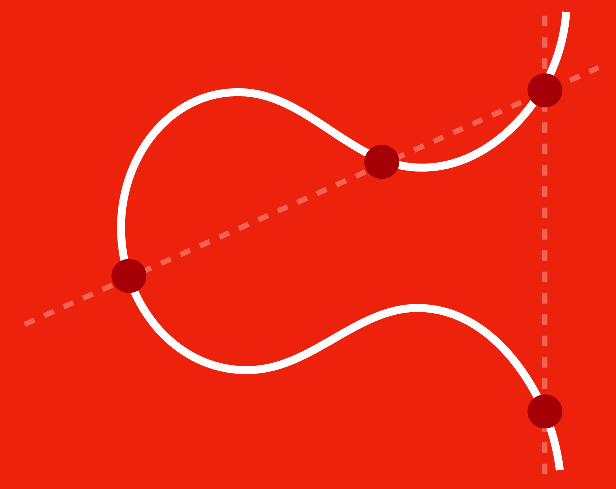
elliptic curves in CSIDH

supersingular elliptic curve

- has $p + 1$ points in $E(\mathbb{F}_p)$
- choose p so that $p + 1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$
- this implies the rational points on E have orders that divide $p + 1$



$$E : y^2 = x^3 + Ax^2 + x, \quad A \in \mathbb{F}_p$$

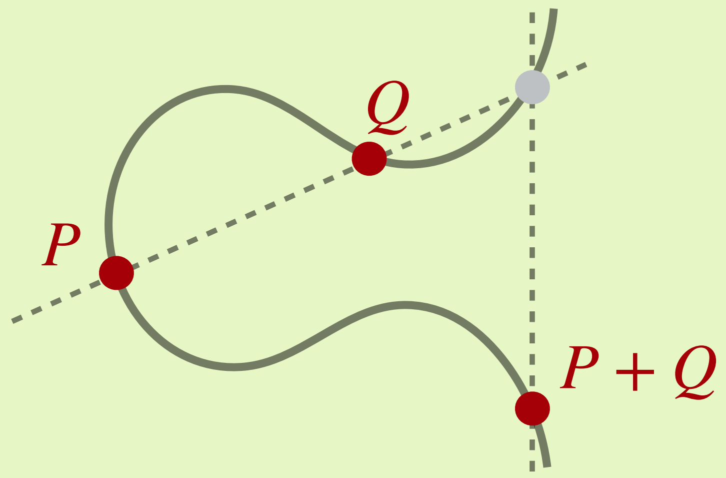


Isogenies & Pairings

elliptic curves in CSIDH

supersingular elliptic curve

- has $p + 1$ points in $E(\mathbb{F}_p)$
- choose p so that $p + 1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$
- this implies the rational points on E have orders that divide $p + 1$



$$E : y^2 = x^3 + Ax^2 + x, \quad A \in \mathbb{F}_p$$

points on such curves

We have that

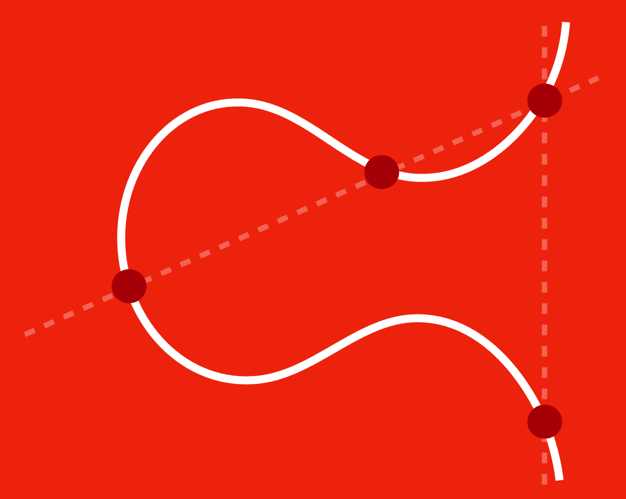
$$E(\mathbb{F}_p) \cong \mathbb{Z}_4 \times \mathbb{Z}_{\ell_1} \times \mathbb{Z}_{\ell_2} \times \ldots \times \mathbb{Z}_{\ell_n},$$

So think of a point $P \in E(\mathbb{F}_p)$ as a sum of points P_i of order ℓ_i

$$P = P_0 + P_1 + P_2 + \ldots + P_n$$

which shows how scalars $[\lambda]$ with $\lambda \in \mathbb{N}$ affect the torsion

$$\begin{aligned} [\ell_2]P &= [\ell_2]P_0 + [\ell_2]P_1 + [\ell_2]P_2 + \ldots + [\ell_2]P_n \\ &= [\ell_2]P_0 + [\ell_2]P_1 + \quad \mathcal{O} \quad + \ldots + [\ell_2]P_n \end{aligned}$$

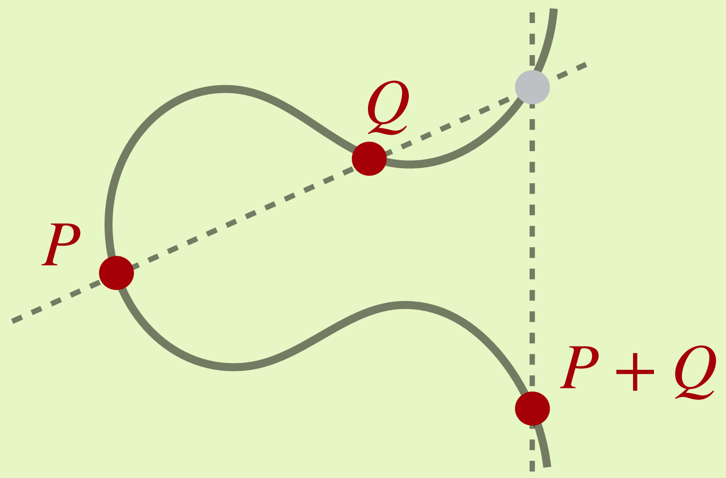


Isogenies & Pairings

elliptic curves in CSIDH

supersingular elliptic curve

- has $p+1$ points in $E(\mathbb{F}_p)$
- choose p so that $p+1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$
- this implies the rational points on E have orders that divide $p+1$



$$E : y^2 = x^3 + Ax^2 + x, \quad A \in \mathbb{F}_p$$

points on such curves

We have that

$$E(\mathbb{F}_p) \cong \mathbb{Z}_4 \times \mathbb{Z}_{\ell_1} \times \mathbb{Z}_{\ell_2} \times \ldots \times \mathbb{Z}_{\ell_n},$$

So think of a point $P \in E(\mathbb{F}_p)$ as a sum of points P_i of order ℓ_i

$$P = P_0 + P_1 + P_2 + \ldots + P_n$$

which shows how scalars $[\lambda]$ with $\lambda \in \mathbb{N}$ affect the torsion

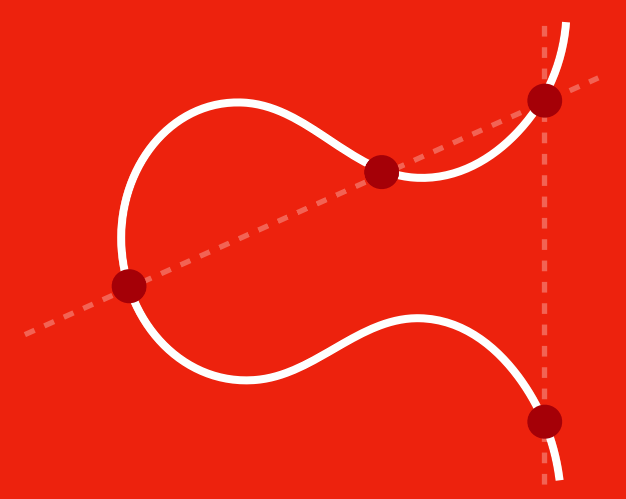
$$\begin{aligned} [\ell_2]P &= [\ell_2]P_0 + [\ell_2]P_1 + [\ell_2]P_2 + \ldots + [\ell_2]P_n \\ &= [\ell_2]P_0 + [\ell_2]P_1 + \quad \mathcal{O} \quad + \ldots + [\ell_2]P_n \end{aligned}$$

the order of P is readable from the non-zero P_i's

the torsion that P is *missing* are precisely the zero P_i's

full-torsion points

we call a point $P \in E(\mathbb{F}_p)$ a **full-torsion point** if the order is $p+1$, equivalently, all P_i are non-zero

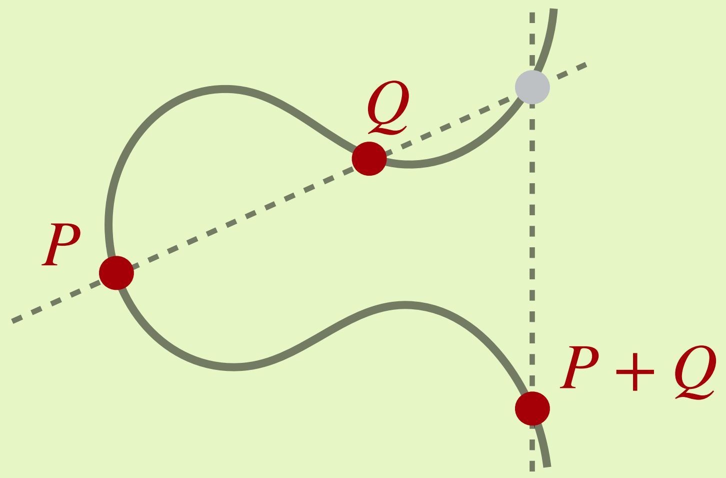


Isogenies & Pairings

elliptic curves in CSIDH

supersingular elliptic curve

- has $p+1$ points in $E(\mathbb{F}_p)$
- choose p so that $p+1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$
- this implies the rational points on E have orders that divide $p+1$



$E : y^2 = x^3 + Ax^2 + x, \quad A \in \mathbb{F}_p$

the order of P is readable from the non-zero P_i's

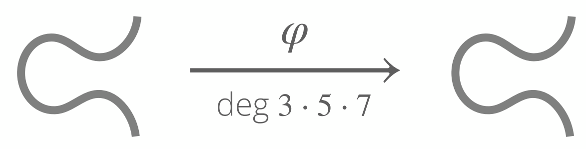
the torsion that P is *missing* are precisely the zero P_i's

full-torsion points

we call a point $P \in E(\mathbb{F}_p)$ a **full-torsion point** if the order is $p+1$, equivalently, all P_i are non-zero

torsion points and isogenies

1. Any* isogeny φ of degree N
   - given by kernel of size N
   - generated by point P of order N



2. Any* isogeny φ of degree $N = \prod \ell_i$
   - splits into sub-isogenies of degree ℓ_i
   - each generated by point P of order ℓ_i

3. Any* isogeny φ of degree $N = \prod \ell_i$
   - computed using one **full-torsion** P
   - per ℓ_i, compute $[\frac{p+1}{\ell_i}]P$ to get $\ker(\varphi_i)$

$P = P_3 + P_5 + P_7 \in E(\mathbb{F}_p)$

$[5 \cdot 7]P = P_3' + \mathcal{O} + \mathcal{O} \in E(\mathbb{F}_p)$

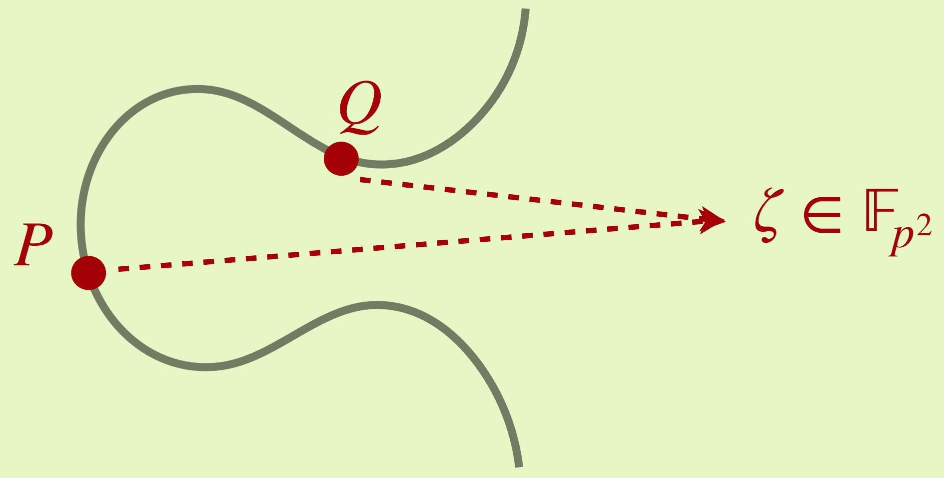
$\varphi_1(P) = \mathcal{O} + P_5' + P_7' \in E'(\mathbb{F}_p)$

*cyclic, separable

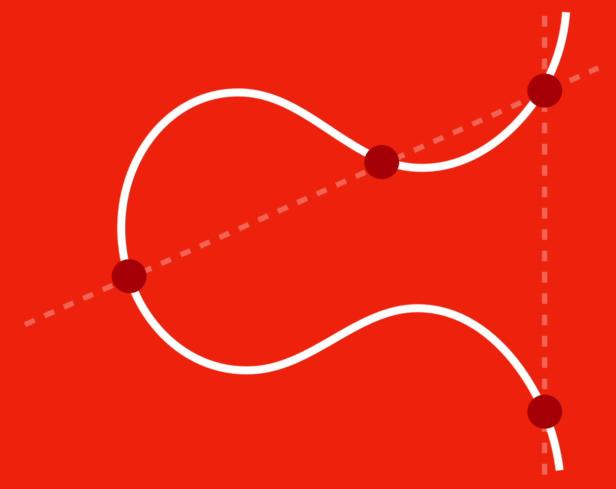
the Tate pairing*

bilinear pairing from torsion groups to fields

- choose a degree r
- take point P of order r on E, that is $P \in E(\mathbb{F}_{p^2})[r]$
- take point Q on E such that $Q \in E(\mathbb{F}_{p^2})/rE(\mathbb{F}_{p^2})$
- then $e_r(P, Q) = \zeta \in \mu_r$



*reduced

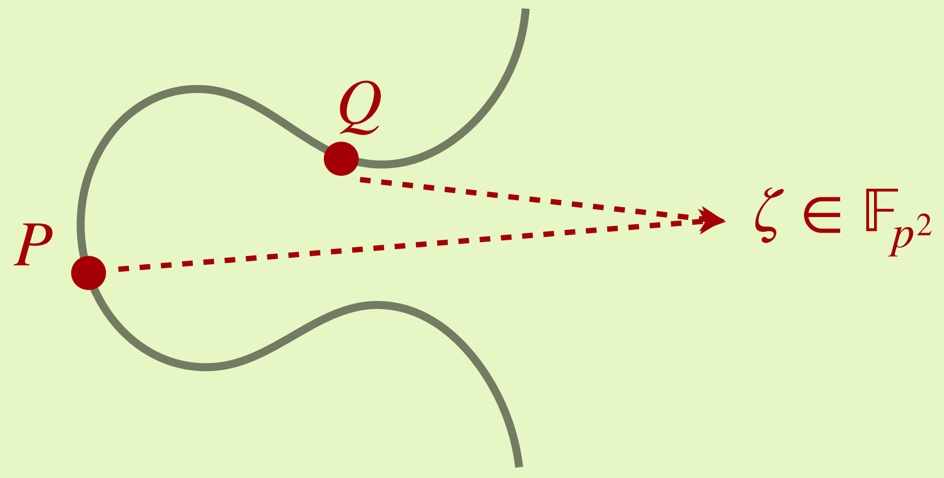


Isogenies & Pairings

the Tate pairing*

bilinear pairing from torsion groups to fields

- choose a degree r
- take point P of order r on E, that is $P \in E(\mathbb{F}_{p^2})[r]$
- take point Q on E such that $Q \in E(\mathbb{F}_{p^2})/rE(\mathbb{F}_{p^2})$
- then $e_r(P, Q) = \zeta \in \mu_r$



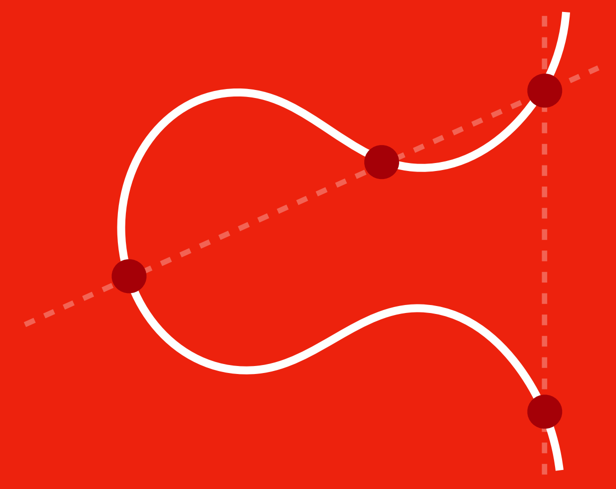
in our specific case

Formally, this pairing is abstract. Specifically in our case, $p+1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$ there is a nice interpretation of this pairing.

Choose r dividing $p+1$, say $r = \prod \ell_i = \frac{p+1}{4}$ then for $P \in E(\mathbb{F}_p)$ we get

$$P = \mathcal{O} + P_1 + P_2 + \ldots + P_n .$$

*reduced

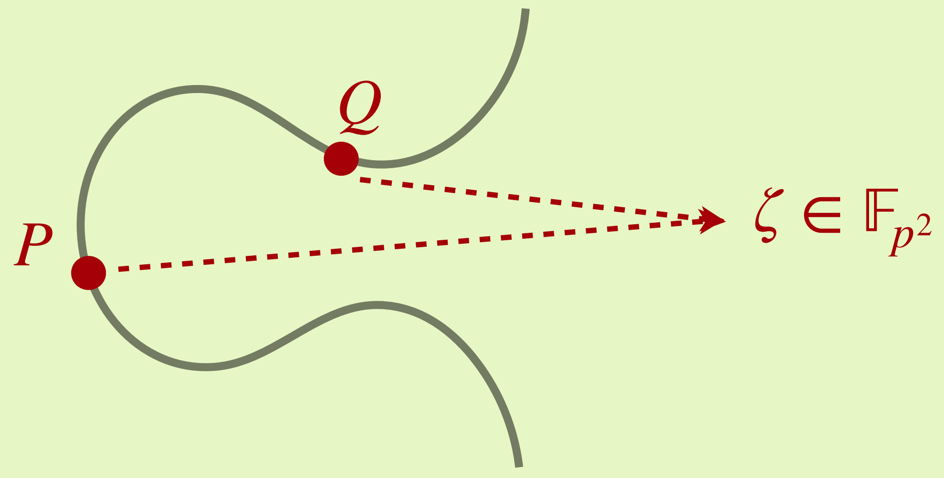


Isogenies & Pairings

the Tate pairing*

bilinear pairing from torsion groups to fields

- choose a degree r
- take point P of order r on E, that is $P \in E(\mathbb{F}_{p^2})[r]$
- take point Q on E such that $Q \in E(\mathbb{F}_{p^2})/rE(\mathbb{F}_{p^2})$
- then $e_r(P, Q) = \zeta \in \mu_r$



in our specific case

Formally, this pairing is abstract. Specifically in our case, $p + 1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$ there is a nice interpretation of this pairing.

Choose r dividing $p+1$, say $r = \prod \ell_i = \frac{p+1}{4}$ then for $P \in E(\mathbb{F}_p)$ we get

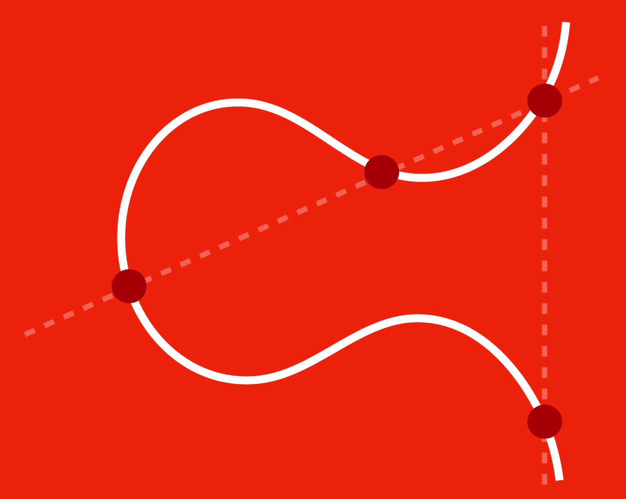
$$P = \mathcal{O} + P_1 + P_2 + \ldots + P_n .$$

For $Q \in E(\mathbb{F}_p)$, we have equivalence by elements R in $rE(\mathbb{F}_{p^2})$. In this scenario, we can think of such elements R as $R_0 + \mathcal{O} + \ldots + \mathcal{O}$, which implies $Q \sim Q'$ whenever

$$Q = Q_0 + Q_1 + Q_2 + \ldots + Q_n \quad \sim \quad Q' = Q_0' + Q_1 + Q_2 + \ldots + Q_n$$

In this specific scenario, we can think of Q as the elements $\mathcal{O} + Q_1 + \ldots + Q_n$

*reduced

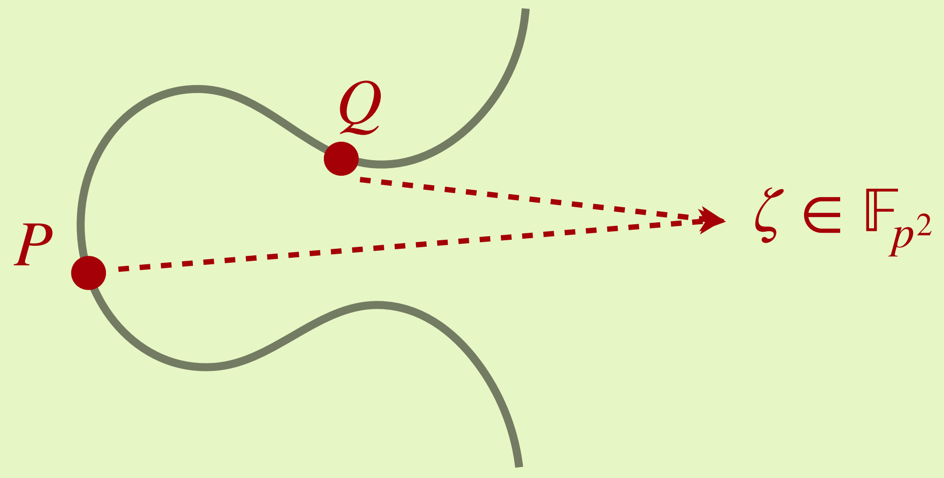


Isogenies & Pairings

the Tate pairing*

bilinear pairing from torsion groups to fields

- choose a degree r
- take point P of order r on E, that is $P \in E(\mathbb{F}_{p^2})[r]$
- take point Q on E such that $Q \in E(\mathbb{F}_{p^2})/rE(\mathbb{F}_{p^2})$
- then $e_r(P, Q) = \zeta \in \mu_r$



in our specific case

Formally, this pairing is abstract. Specifically in our case, $p + 1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$ there is a nice interpretation of this pairing.

Choose r dividing $p+1$, say $r = \prod \ell_i = \dfrac{p+1}{4}$ then for $P \in E(\mathbb{F}_p)$ we get

$$P = \mathcal{O} + P_1 + P_2 + \ldots + P_n .$$

For $Q \in E(\mathbb{F}_p)$, we have equivalence by elements R in $rE(\mathbb{F}_{p^2})$. In this scenario, we can think of such elements R as $R_0 + \mathcal{O} + \ldots + \mathcal{O}$, which implies $Q \sim Q'$ whenever

$$Q = Q_0 + Q_1 + Q_2 + \ldots + Q_n \quad \sim \quad Q' = Q_0' + Q_1 + Q_2 + \ldots + Q_n$$

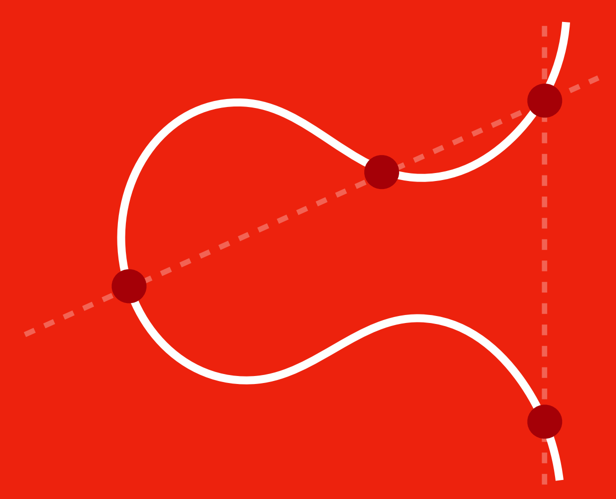
In this specific scenario, we can think of Q as the elements $\mathcal{O} + Q_1 + \ldots + Q_n$

!

problem

If we pick $P, Q \in E(\mathbb{F}_p)$, then Q is a multiple of P. Then $e_r(P, Q) = 1$

*reduced

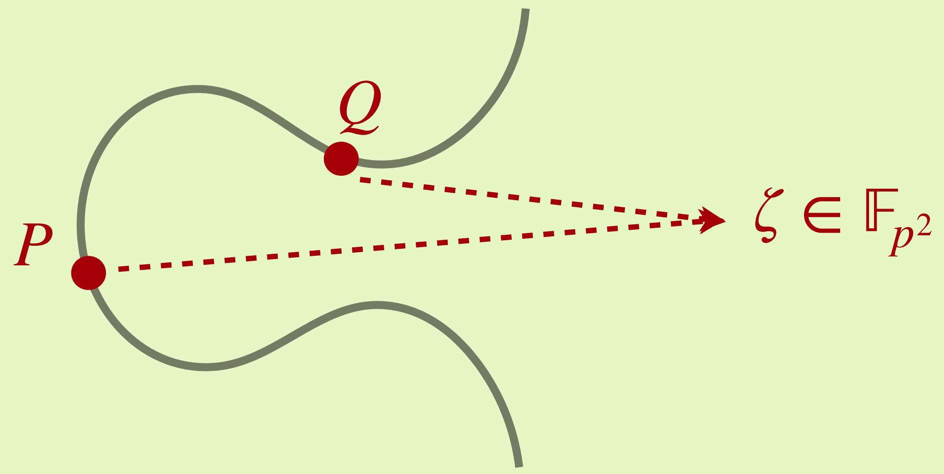


Isogenies & Pairings

the Tate pairing*

bilinear pairing from torsion groups to fields

- choose a degree r
- take point P of order r on E, that is $P \in E(\mathbb{F}_{p^2})[r]$
- take point Q on E such that $Q \in E(\mathbb{F}_{p^2})/rE(\mathbb{F}_{p^2})$
- then $e_r(P,Q) = \zeta \in \mu_r$



in our specific case

Formally, this pairing is abstract. Specifically in our case, $p+1 = 4 \cdot \ell_1 \cdot \ell_2 \cdot \ldots \cdot \ell_n$ there is a nice interpretation of this pairing.

Choose r dividing $p+1$, say $r = \prod \ell_i = \frac{p+1}{4}$ then for $P \in E(\mathbb{F}_p)$ we get

$$P = \mathcal{O} + P_1 + P_2 + \ldots + P_n.$$

For $Q \in E(\mathbb{F}_p)$, we have equivalence by elements R in $rE(\mathbb{F}_{p^2})$. In this scenario, we can think of such elements R as $R_0 + \mathcal{O} + \ldots + \mathcal{O}$, which implies $Q \sim Q'$ whenever

$$Q = Q_0 + Q_1 + Q_2 + \ldots + Q_n \sim Q' = Q_0' + Q_1 + Q_2 + \ldots + Q_n$$

In this specific scenario, we can think of Q as the elements $\mathcal{O} + Q_1 + \ldots + Q_n$

! **problem**

If we pick $P, Q \in E(\mathbb{F}_p)$, then Q is a multiple of P. Then $e_r(P,Q) = 1$

✓ **solution**

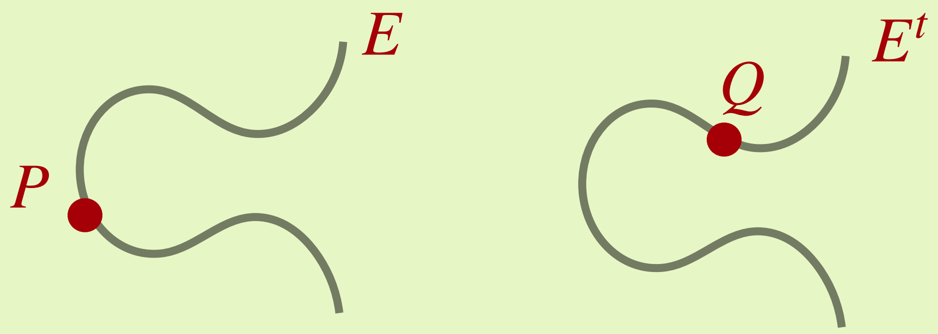
Work over $E[r] \subseteq E(\mathbb{F}_{p^2})$. In our specific case, just use Q on the *twist*

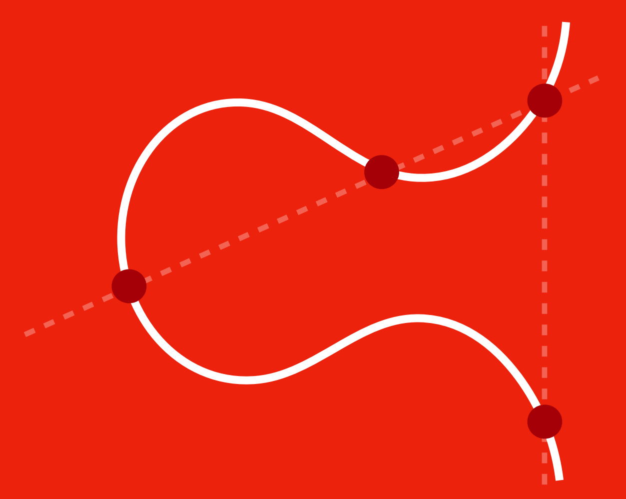
*reduced

the twist of E

Twist over $\mathbb{F}_p$ of supersingular curve E

- a curve E^t with $p + 1$ points over $\mathbb{F}_p$
- isomorphic to a specific subset of $E(\mathbb{F}_{p^2})$
- used in CSIDH to "move backwards" in graph
- want $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, both full order



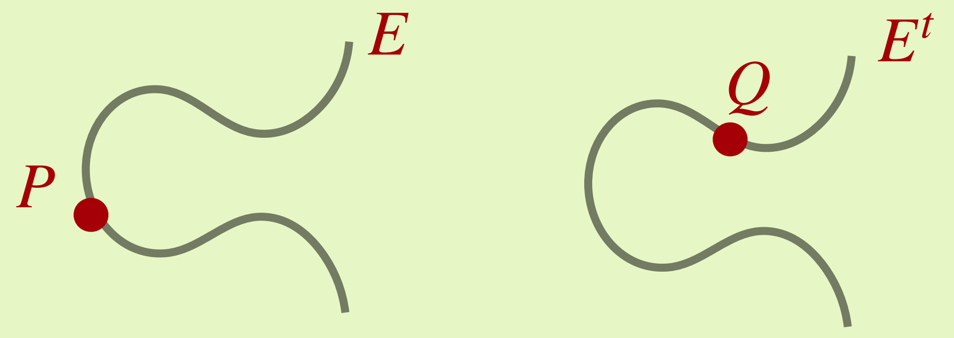


Isogenies & Pairings

the twist of E

Twist over $\mathbb{F}_p$ of supersingular curve E

- a curve E^t with $p+1$ points over $\mathbb{F}_p$
- isomorphic to a specific subset of $E(\mathbb{F}_{p^2})$
- used in CSIDH to "move backwards" in graph
- want $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, both full order



1

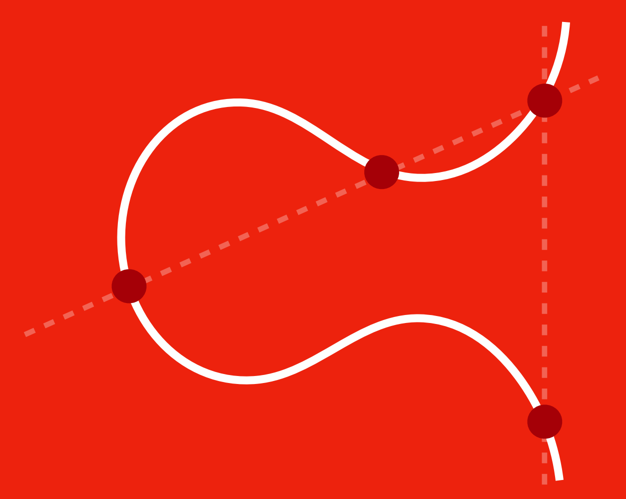
consider P and Q as

$$P = P_0 + P_1 + \ldots + P_n$$
$$Q = Q_0 + Q_1 + \ldots + Q_n$$

2

let $r = p + 1$

Tate pairing $e_r(P, Q)$ captures where **both** $P_i, Q_i \neq \mathcal{O}$

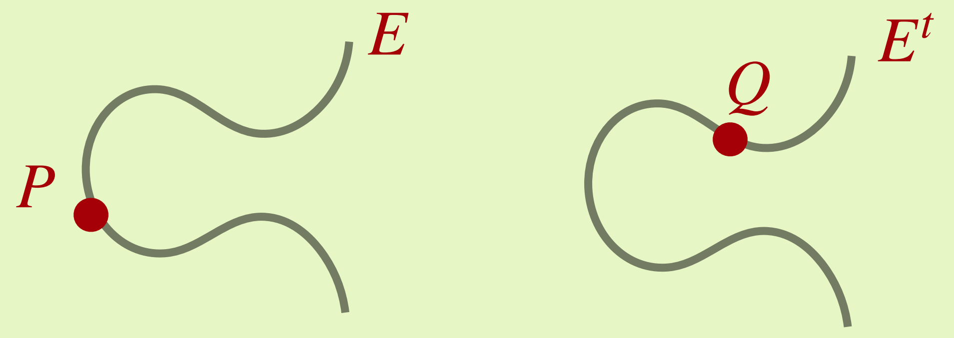


Isogenies & Pairings

the twist of E

Twist over $\mathbb{F}_p$ of supersingular curve E

- a curve E^t with $p+1$ points over $\mathbb{F}_p$
- isomorphic to a specific subset of $E(\mathbb{F}_{p^2})$
- used in CSIDH to “move backwards” in graph
- want $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, both full order



1

consider P and Q as

$$P = P_0 + P_1 + \ldots + P_n$$

$$Q = Q_0 + Q_1 + \ldots + Q_n$$

2

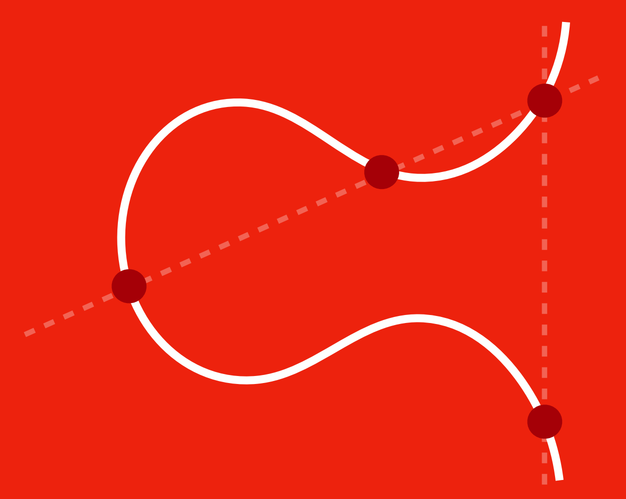
let $r = p+1$

Tate pairing $e_r(P, Q)$ captures where **both** $P_i, Q_i \neq \mathcal{O}$

crucial lemma

Let $P \in E(\mathbb{F}_p)$, $Q \in E^t(\mathbb{F}_p)$, and $r = p+1$. Let $\zeta = e_r(P, Q) \in \mathbb{F}_{p^2}$.

Then ζ is an r-th root of unity, whose order is precisely $\gcd$ of order of P, order of Q

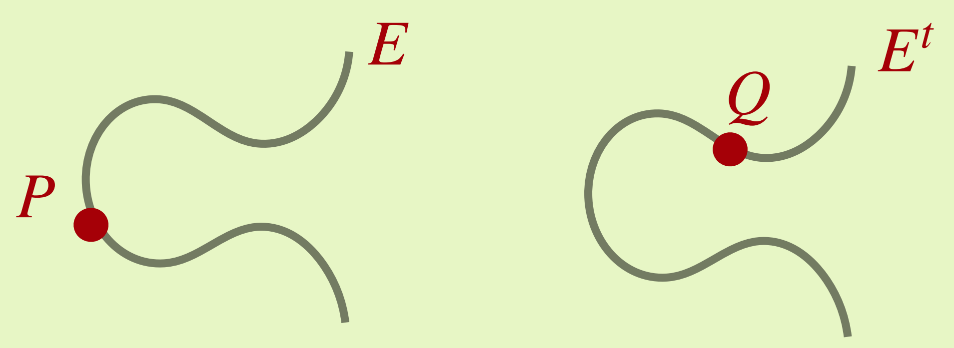


Isogenies & Pairings

the twist of E

Twist over $\mathbb{F}_p$ of supersingular curve E

- a curve E^t with $p+1$ points over $\mathbb{F}_p$
- isomorphic to a specific subset of $E(\mathbb{F}_{p^2})$
- used in CSIDH to "move backwards" in graph
- want $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, both full order



1

consider P and Q as

$$P = P_0 + P_1 + \ldots + P_n$$
$$Q = Q_0 + Q_1 + \ldots + Q_n$$

2

let $r = p+1$

Tate pairing $e_r(P, Q)$ captures where **both** $P_i, Q_i \neq \mathcal{O}$

crucial lemma

Let $P \in E(\mathbb{F}_p)$, $Q \in E^t(\mathbb{F}_p)$, and $r = p+1$. Let $\zeta = e_r(P, Q) \in \mathbb{F}_{p^2}$.

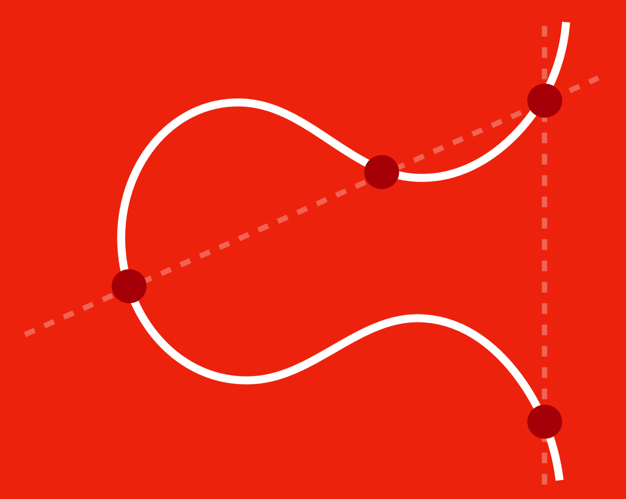
Then ζ is an r-th root of unity, whose order is precisely $\gcd$ of order of P, order of Q

example

If P and Q both full torsion, then ζ has order $r = p+1$

example

If P has order 5, and Q has order 15, then ζ has order 5

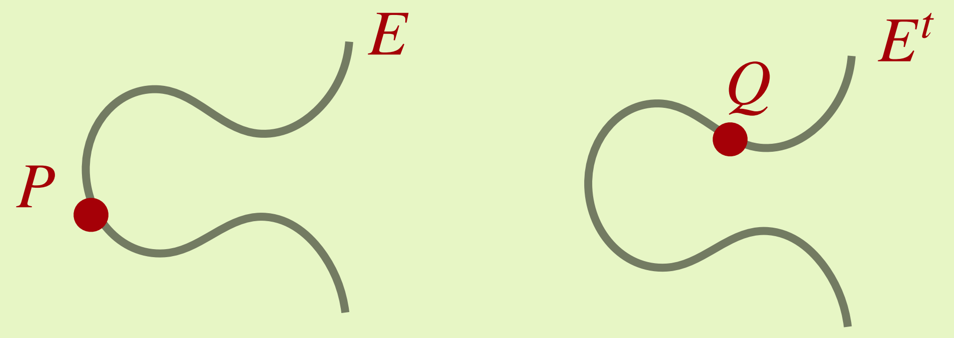


Isogenies & Pairings

the twist of E

Twist over $\mathbb{F}_p$ of supersingular curve E

- a curve E^t with $p+1$ points over $\mathbb{F}_p$
- isomorphic to a specific subset of $E(\mathbb{F}_{p^2})$
- used in CSIDH to "move backwards" in graph
- want $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, both full order



1

consider P and Q as

$$P = P_0 + P_1 + \ldots + P_n$$

$$Q = Q_0 + Q_1 + \ldots + Q_n$$

2

let $r = p + 1$

Tate pairing $e_r(P, Q)$ captures where **both** $P_i, Q_i \neq \mathcal{O}$

crucial lemma

Let $P \in E(\mathbb{F}_p)$, $Q \in E^t(\mathbb{F}_p)$, and $r = p+1$. Let $\zeta = e_r(P, Q) \in \mathbb{F}_{p^2}$.

Then ζ is an r-th root of unity, whose order is precisely $\gcd$ of order of P, order of Q

example

If P and Q both full torsion, then ζ has order $r = p + 1$

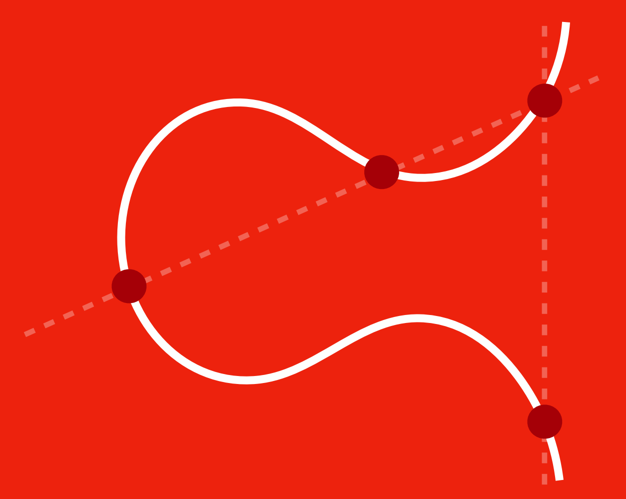
example

If P has order 5, and Q has order 15, then ζ has order 5

!

notice

Curve arithmetic is slow!
Field arithmetic is fast!!
(more than factor 6)

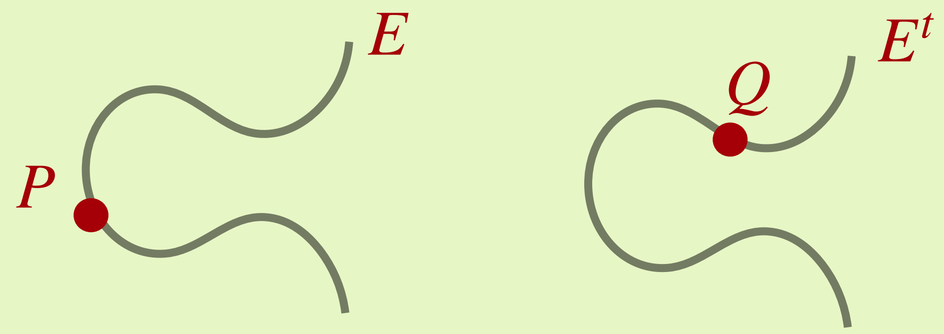


Isogenies & Pairings

the twist of E

Twist over $\mathbb{F}_p$ of supersingular curve E

- a curve E^t with $p+1$ points over $\mathbb{F}_p$
- isomorphic to a specific subset of $E(\mathbb{F}_{p^2})$
- used in CSIDH to "move backwards" in graph
- want $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, both full order



1

consider P and Q as

$$P = P_0 + P_1 + \ldots + P_n$$
$$Q = Q_0 + Q_1 + \ldots + Q_n$$

2

let $r = p + 1$

Tate pairing $e_r(P, Q)$ captures where **both** $P_i, Q_i \neq \mathcal{O}$

crucial lemma

Let $P \in E(\mathbb{F}_p)$, $Q \in E^t(\mathbb{F}_p)$, and $r = p + 1$. Let $\zeta = e_r(P, Q) \in \mathbb{F}_{p^2}$.

Then ζ is an r-th root of unity, whose order is precisely $\gcd$ of order of P, order of Q

example

If P and Q both full torsion, then ζ has order $r = p + 1$

example

If P has order 5, and Q has order 15, then ζ has order 5

!

notice

Curve arithmetic is slow!
Field arithmetic is fast!!
(more than factor 6)

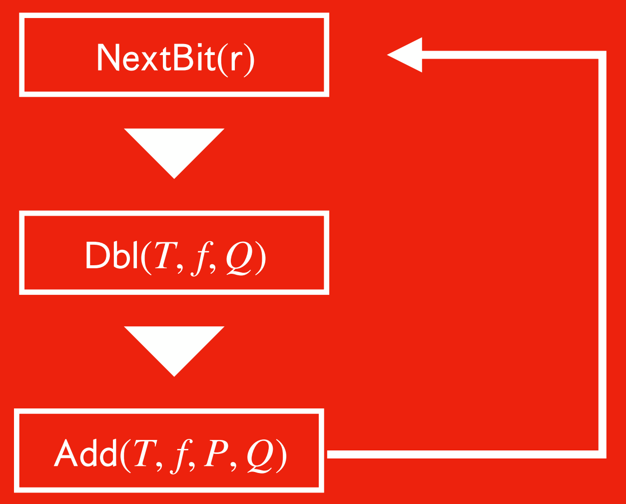
✓

core idea

Pick random $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$
Instead of using curve arithmetic to compute their orders, use ζ to compute the overlap in orders!

Pairings are quite slow

2

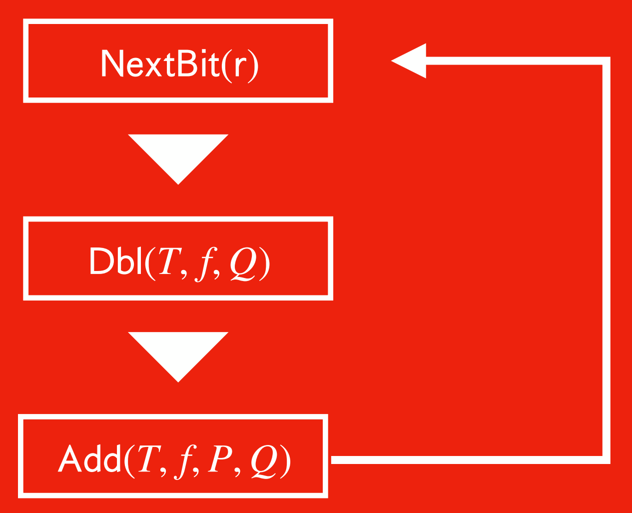


Speeding-up general pairings

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

2



Speeding-up general pairings

pairing crypto

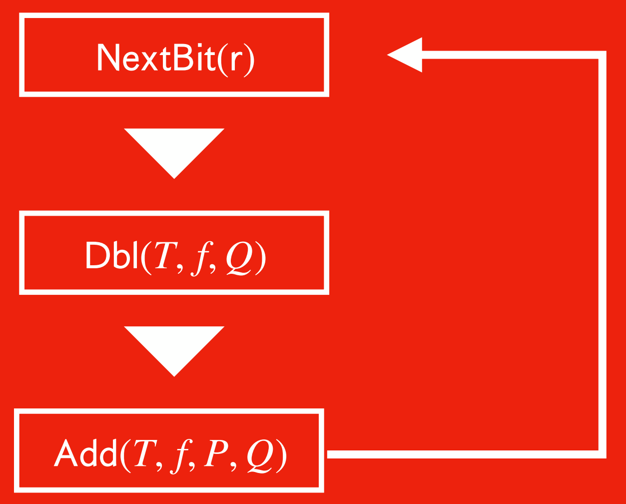
Choose a “nice” curve E,
Choose a “nice” prime p,
to do **pairings** with

Computing $e(P, Q)$
is quite **fast**!

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don’t use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

2



Speeding-up general pairings

pairing crypto

Choose a “nice” curve E,
Choose a “nice” prime p,
to do **pairings** with

Computing $e(P, Q)$
is quite **fast**!

isogeny crypto

Choose a “nice” curve E,
Choose a “nice” prime p,
to do **isogenies** with

These are mediocre curves,
and definitely bad primes,
to do **pairings** with

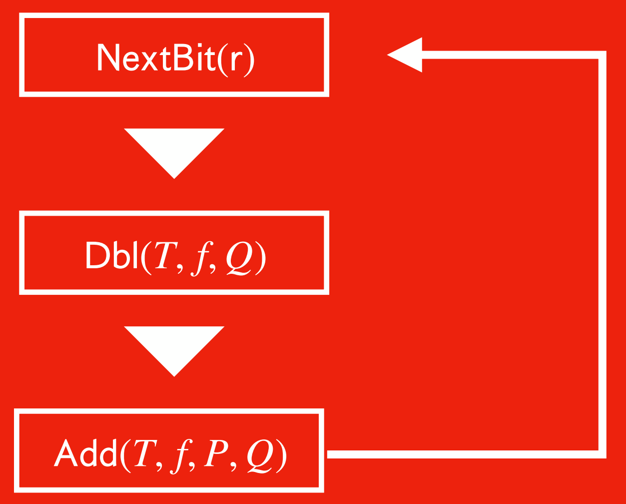
Computing $e(P, Q)$
seems way too **slow**!

✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don’t use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

2



Speeding-up general pairings

pairing crypto

Choose a "nice" curve E,
Choose a "nice" prime p,
to do **pairings** with

Computing $e(P, Q)$
is quite **fast**!

isogeny crypto

Choose a "nice" curve E,
Choose a "nice" prime p,
to do **isogenies** with

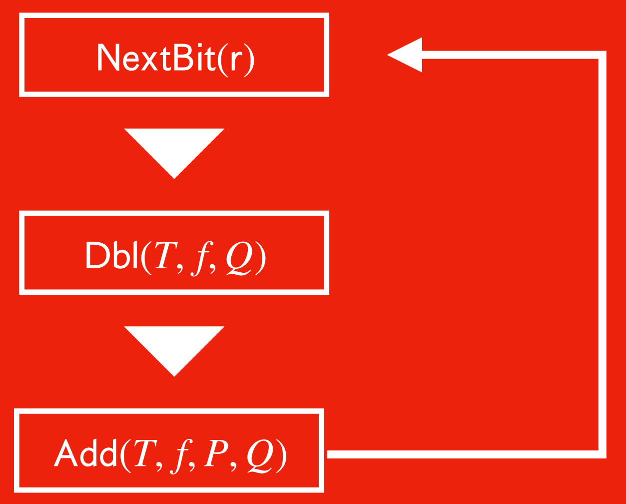
These are mediocre curves,
and definitely bad primes,
to do **pairings** with

Computing $e(P, Q)$
seems way too **slow**!

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don't use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

2



Speeding-up general pairings

pairing crypto

Choose a "nice" curve E,
Choose a "nice" prime p,
to do **pairings** with

Computing $e(P, Q)$
is quite **fast**!

💔

isogeny crypto

Choose a "nice" curve E,
Choose a "nice" prime p,
to do **isogenies** with

These are mediocre curves,
and definitely bad primes,
to do **pairings** with

Computing $e(P, Q)$
seems way too **slow**!

✓ **core idea**

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don't use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

MAIN RESULTS

1 make pairings great again

2 apply **core idea**

3 faster isogeny algorithms!

2

NextBit(r)

Dbl(T, f, Q)

Add(T, f, P, Q)

Speeding-up general pairings

pairing crypto

Choose a “nice” curve E,
Choose a “nice” prime p,
to do **pairings** with

Computing $e(P, Q)$
is quite **fast**!

isogeny crypto

Choose a “nice” curve E,
Choose a “nice” prime p,
to do **isogenies** with

These are mediocre curves,
and definitely bad primes,
to do **pairings** with

Computing $e(P, Q)$
seems way too **slow**!

✓ **core idea**

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don’t use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

MAIN RESULTS

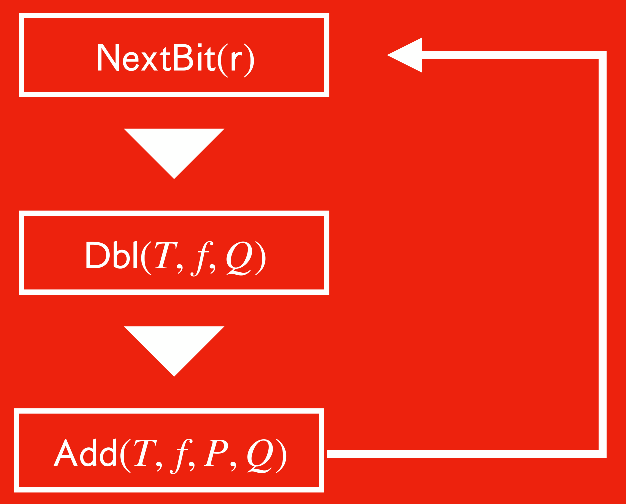
1 make pairings great again

2 apply **core idea**

3 faster isogeny algorithms!

first this

2



Speeding-up general pairings

pairing crypto

Choose a "nice" curve E,
Choose a "nice" prime p,
to do **pairings** with

Computing $e(P, Q)$
is quite **fast**!

isogeny crypto

Choose a "nice" curve E,
Choose a "nice" prime p,
to do **isogenies** with

These are mediocre curves,
and definitely bad primes,
to do **pairings** with

Computing $e(P, Q)$
seems way too **slow**!

✓ **core idea**

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don't use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

MAIN RESULTS

1 make pairings great again

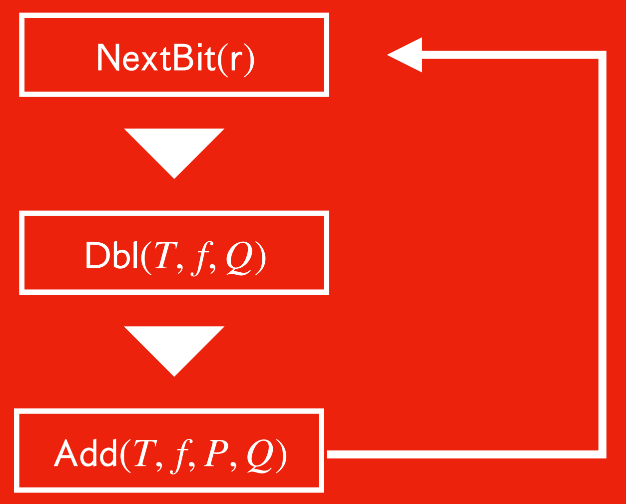
2 apply **core idea**

3 faster isogeny algorithms!

first this

then this

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

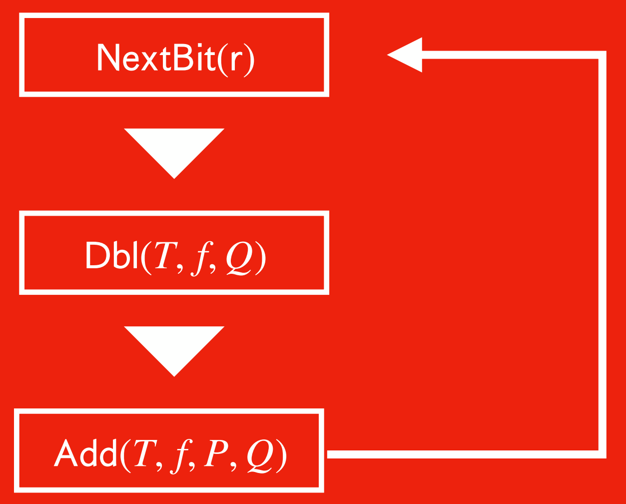
For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

general approach

Instead I describe the general approach, and leave all details out

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

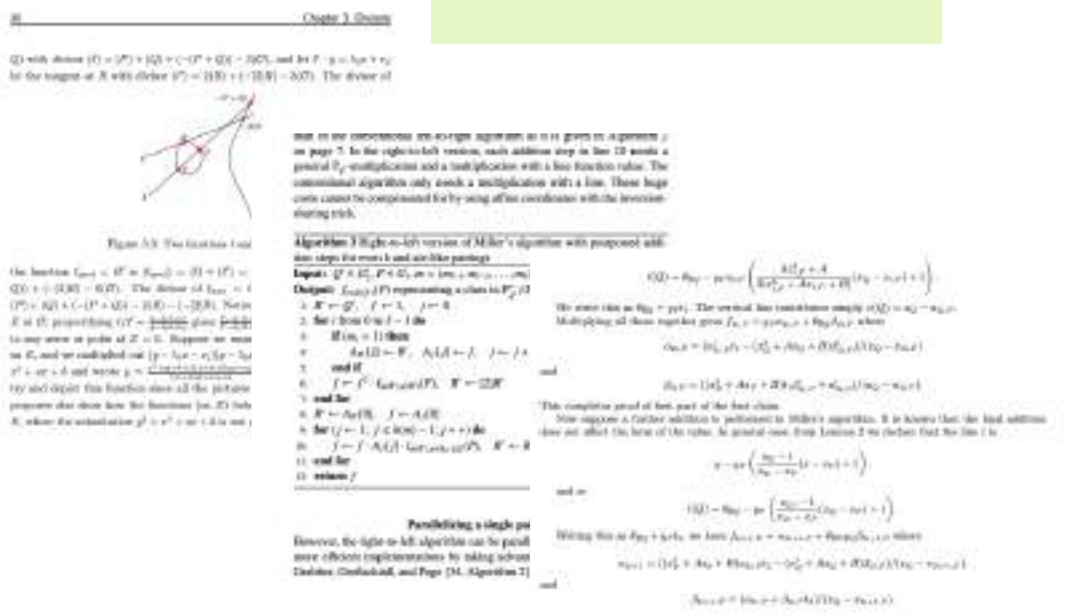
✓

general approach

Instead I describe the general approach, and leave all details out

0

take some literature



1

implement all tricks that apply

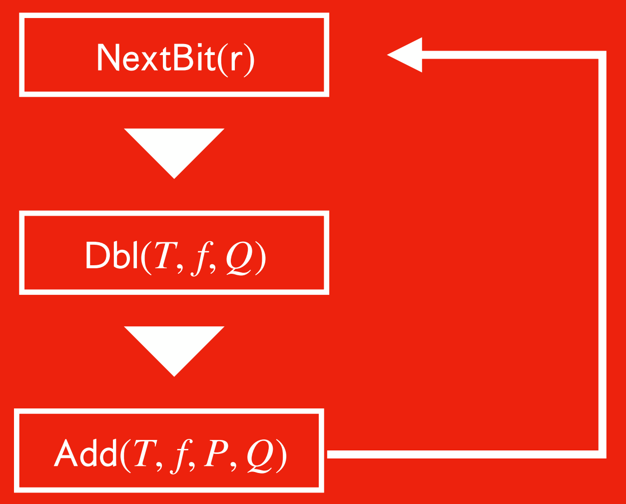
2

benchmark speed and finetune

3

fast pairings

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

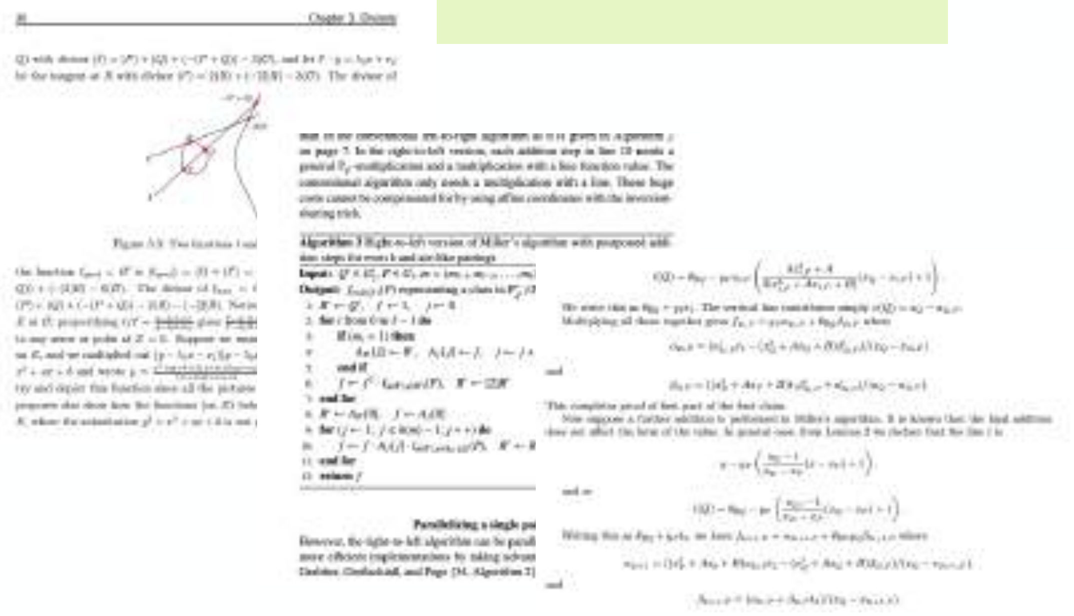
✓ **core idea**

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓ **general approach**

Instead I describe the general approach, and leave all details out

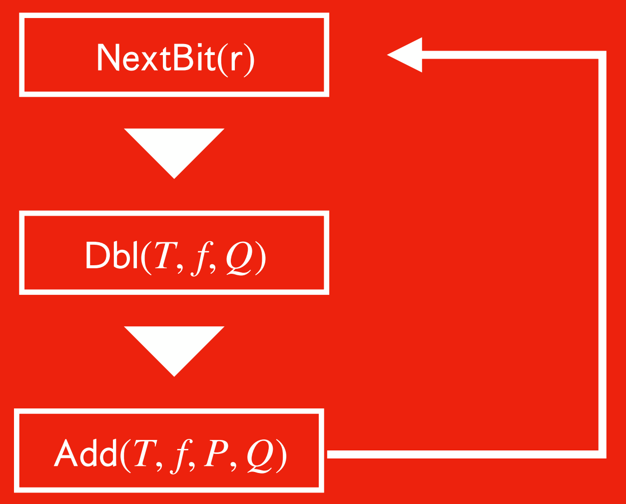
0 take some literature



1 implement all tricks that apply

2 benchmark speed and finetune

3 fast pairings



Speeding-up general pairings

! **general notice**

Computing pairings fast is quite technical.
Better suited for papers than slides

✓ **core idea**

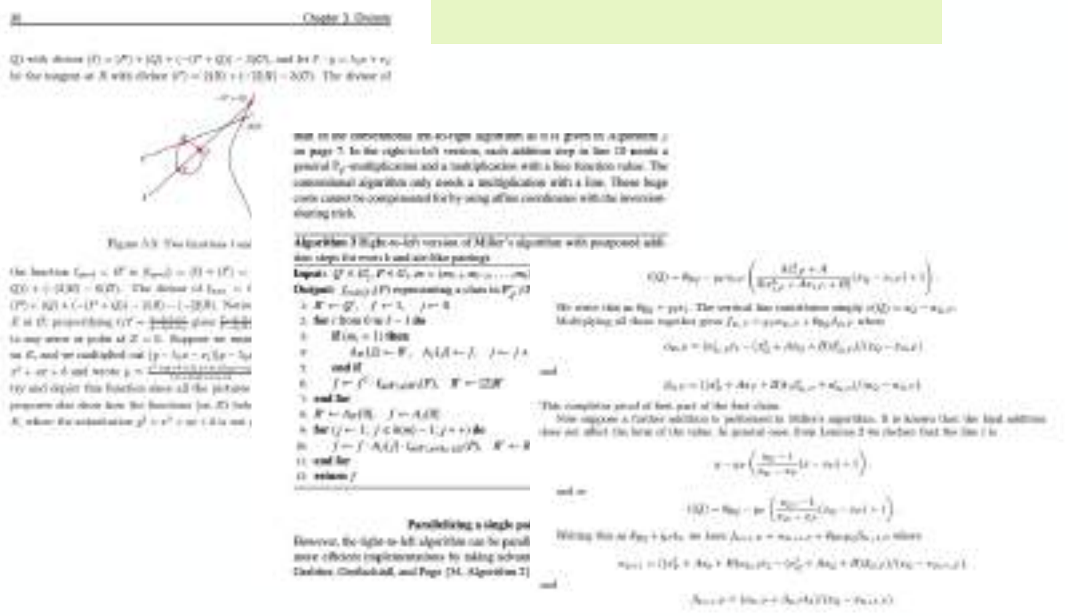
For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓ **general approach**

Instead I describe the general approach, and leave all details out

---

0 take some literature

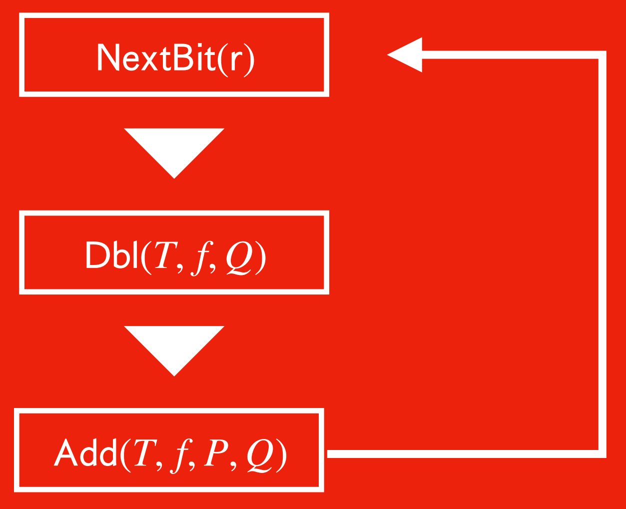


1 implement all tricks that apply

2 benchmark speed and finetune

3 fast pairings

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

general approach

Instead I describe the general approach,
and leave all details out

✓

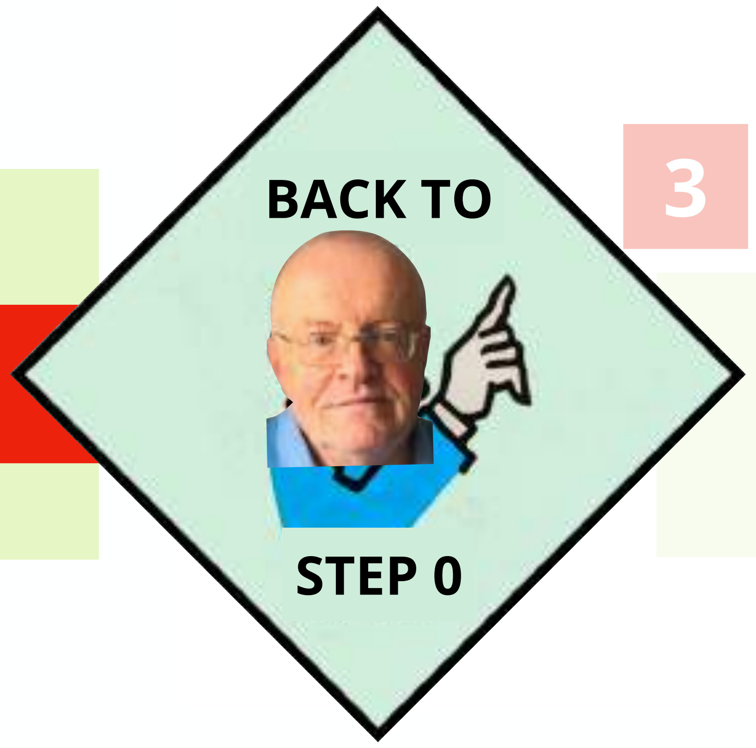
core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don't use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

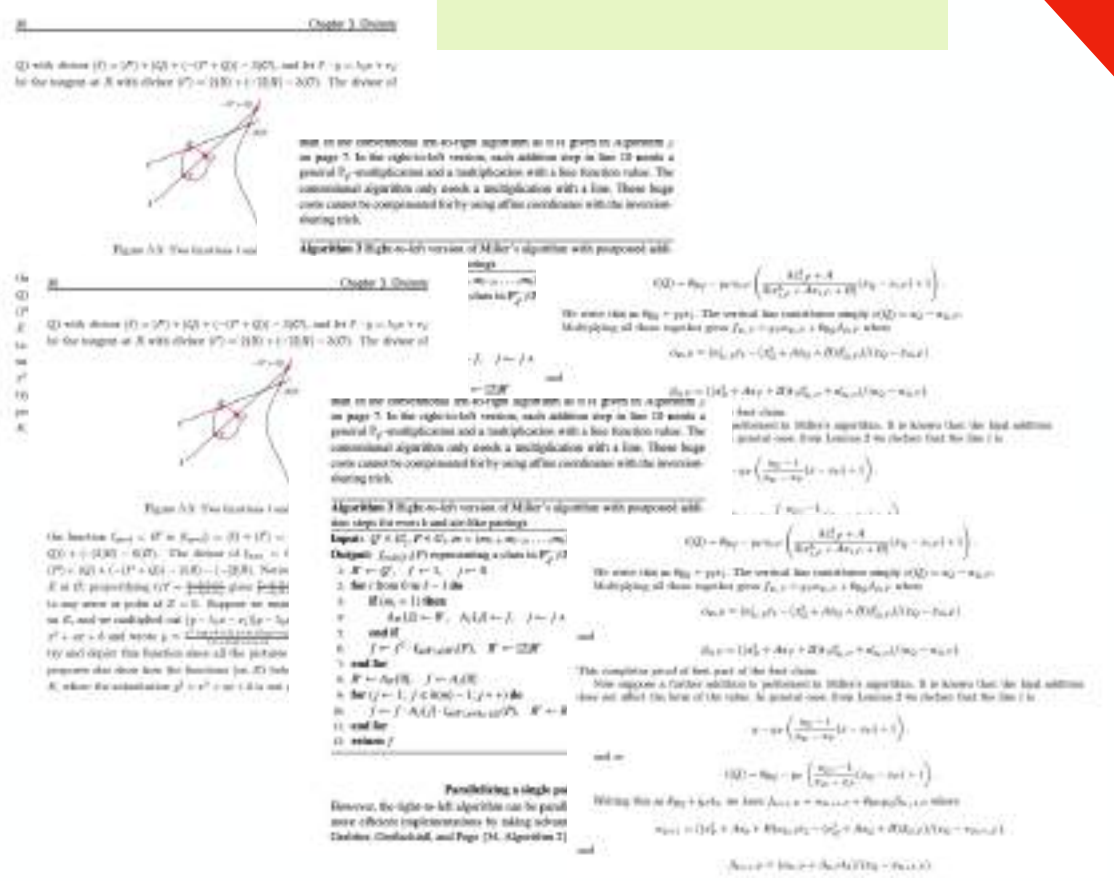
0 take some literature

1

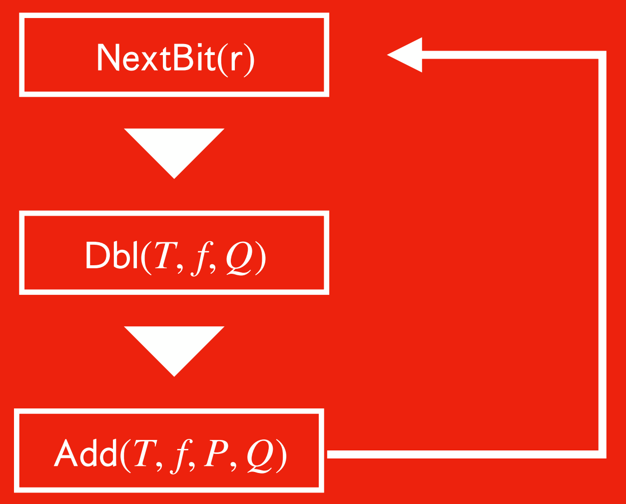
2



3 fast pairings



2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

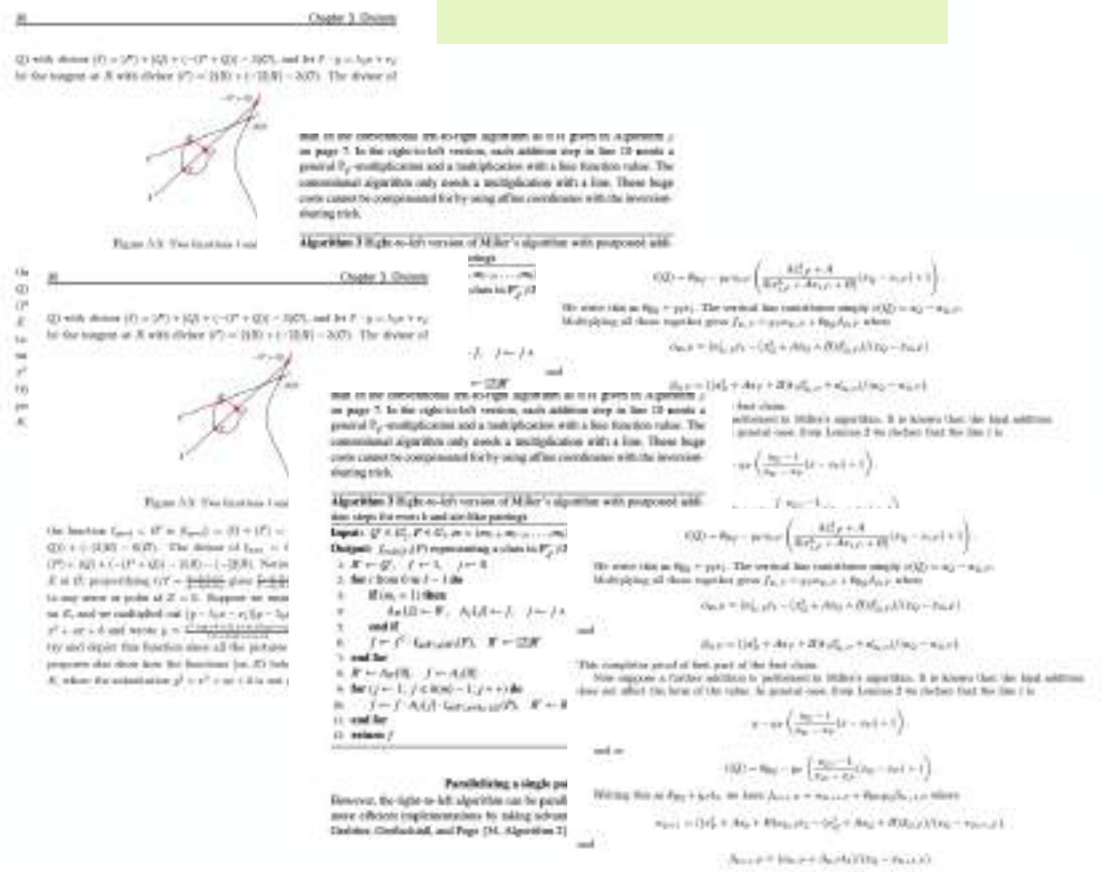
For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

general approach

Instead I describe the general approach, and leave all details out

0 take some literature

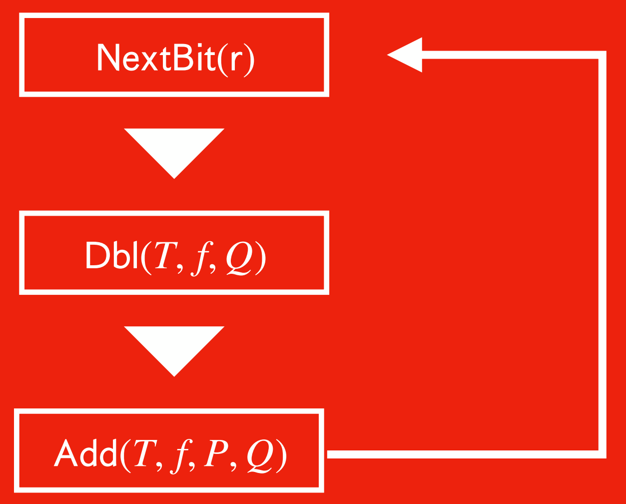


1 implement all tricks that apply

2 benchmark speed and finetune

3 fast pairings

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

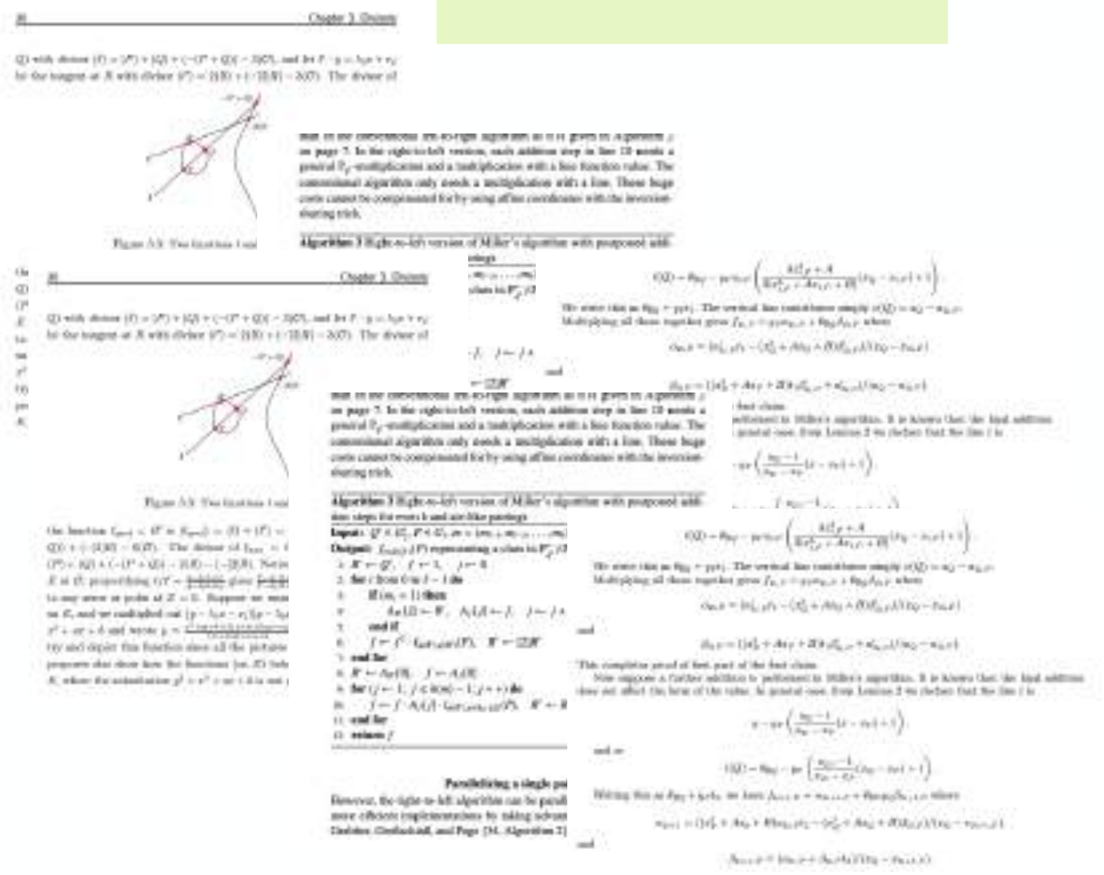
For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

general approach

Instead I describe the general approach, and leave all details out

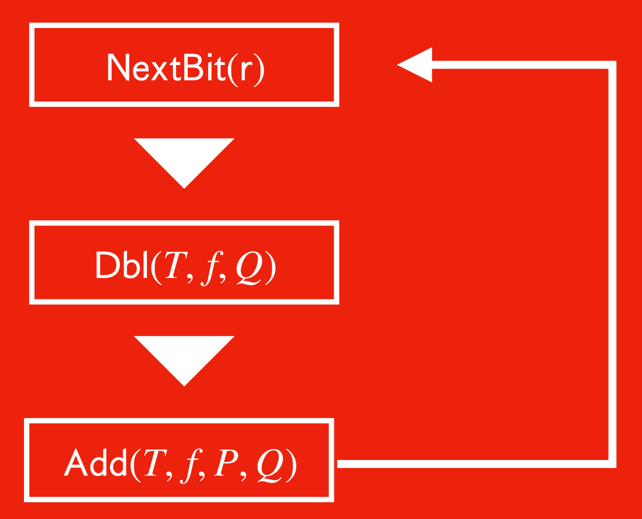
0 take some literature



1 implement all tricks that apply

2 benchmark speed and finetune

3 fast pairings



Speeding-up general pairings

! **general notice**

Computing pairings fast is quite technical.
Better suited for papers than slides

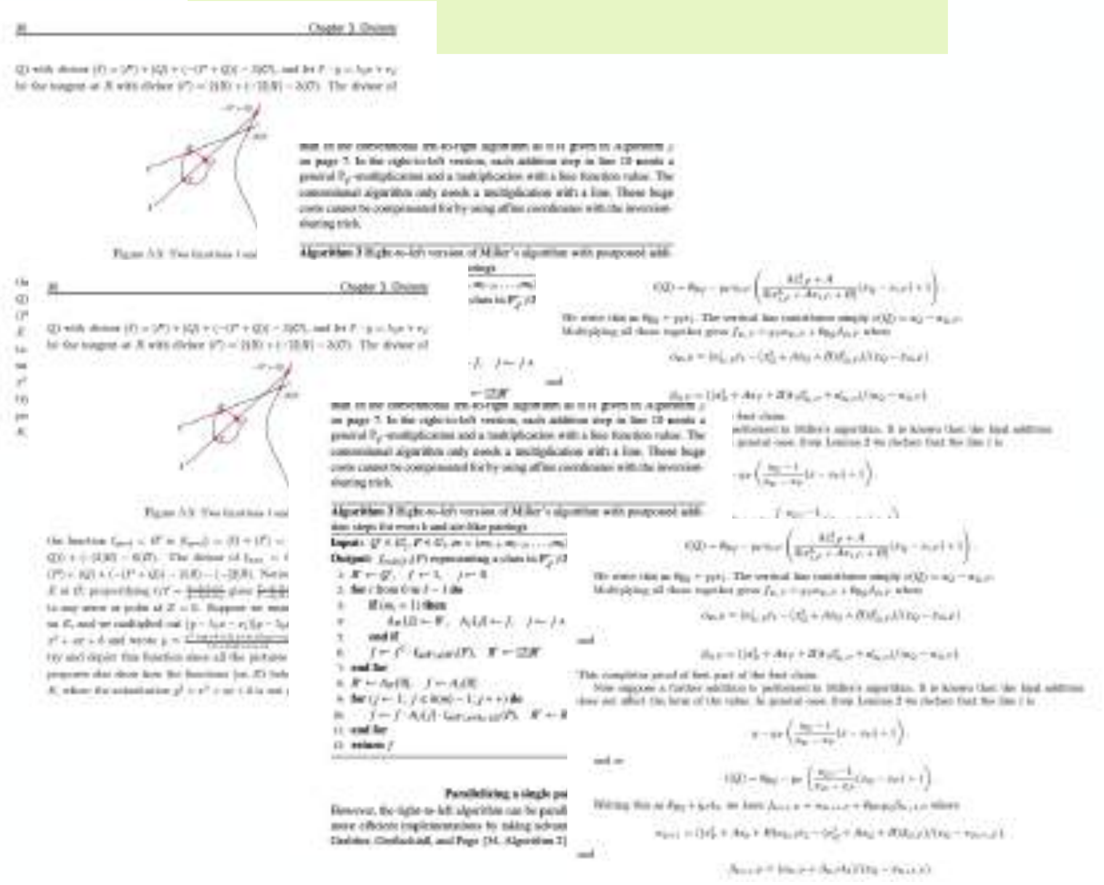
✓ **core idea**

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓ **general approach**

Instead I describe the general approach, and leave all details out

0 take some literature

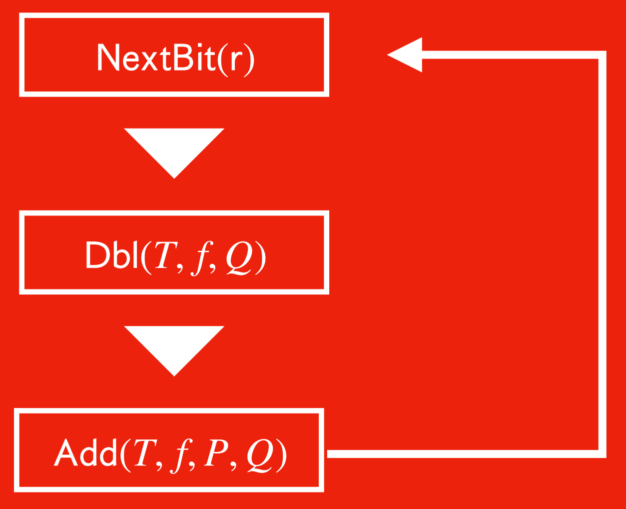


1 implement all tricks that apply

2 benchmark speed and finetune

3 fast pairings

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

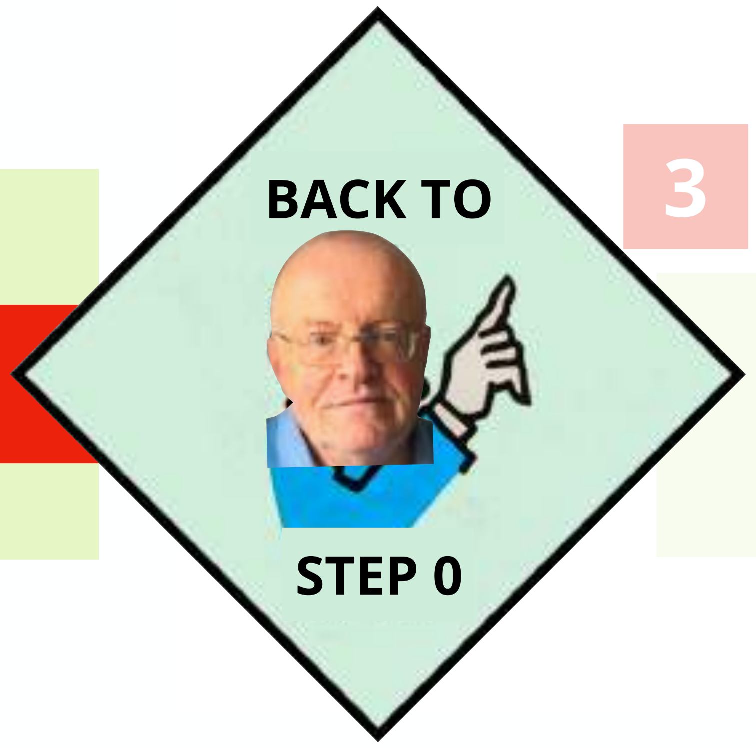
general approach

Instead I describe the general approach, and leave all details out

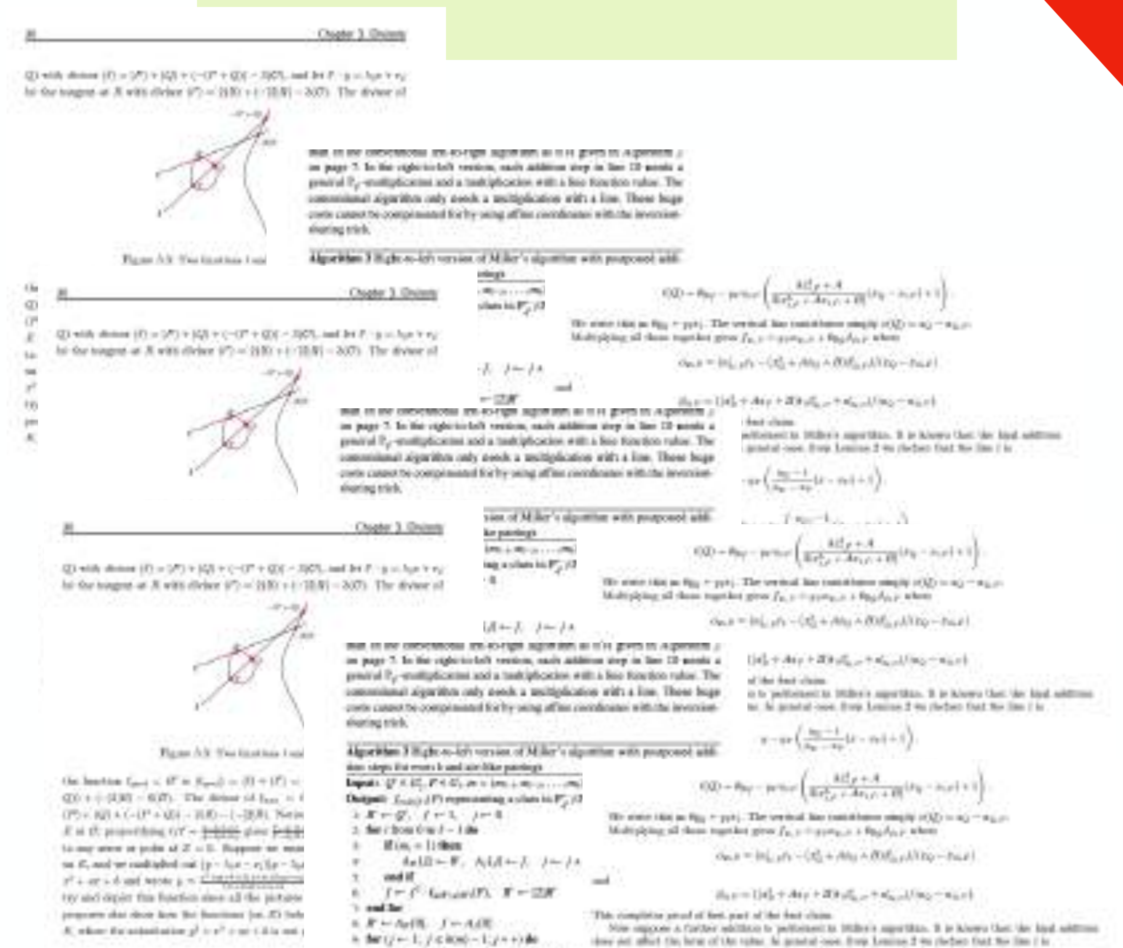
0 take some literature

1

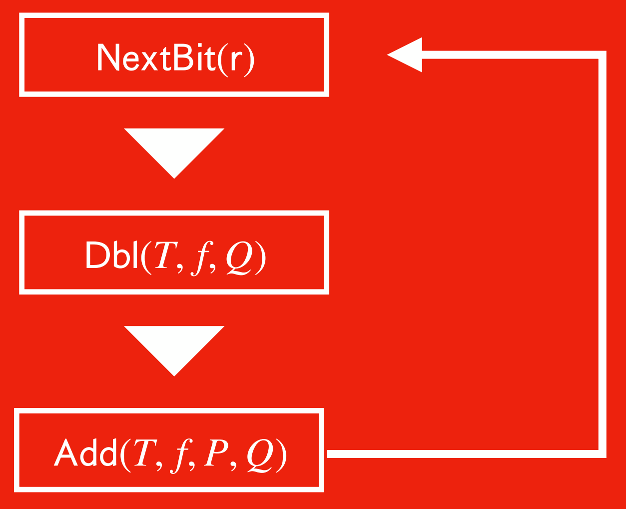
2



3 fast pairings



2



Speeding-up general pairings

! **general notice**

Computing pairings fast is quite technical.
Better suited for papers than slides

✓ **core idea**

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P,Q)$ to get overlap in orders!

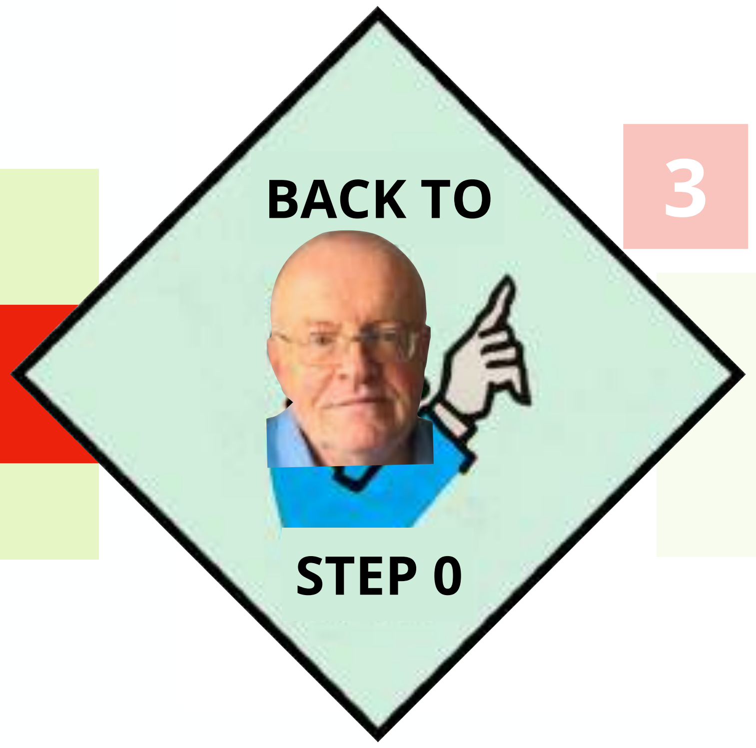
✓ **general approach**

Instead I describe the general approach, and leave all details out

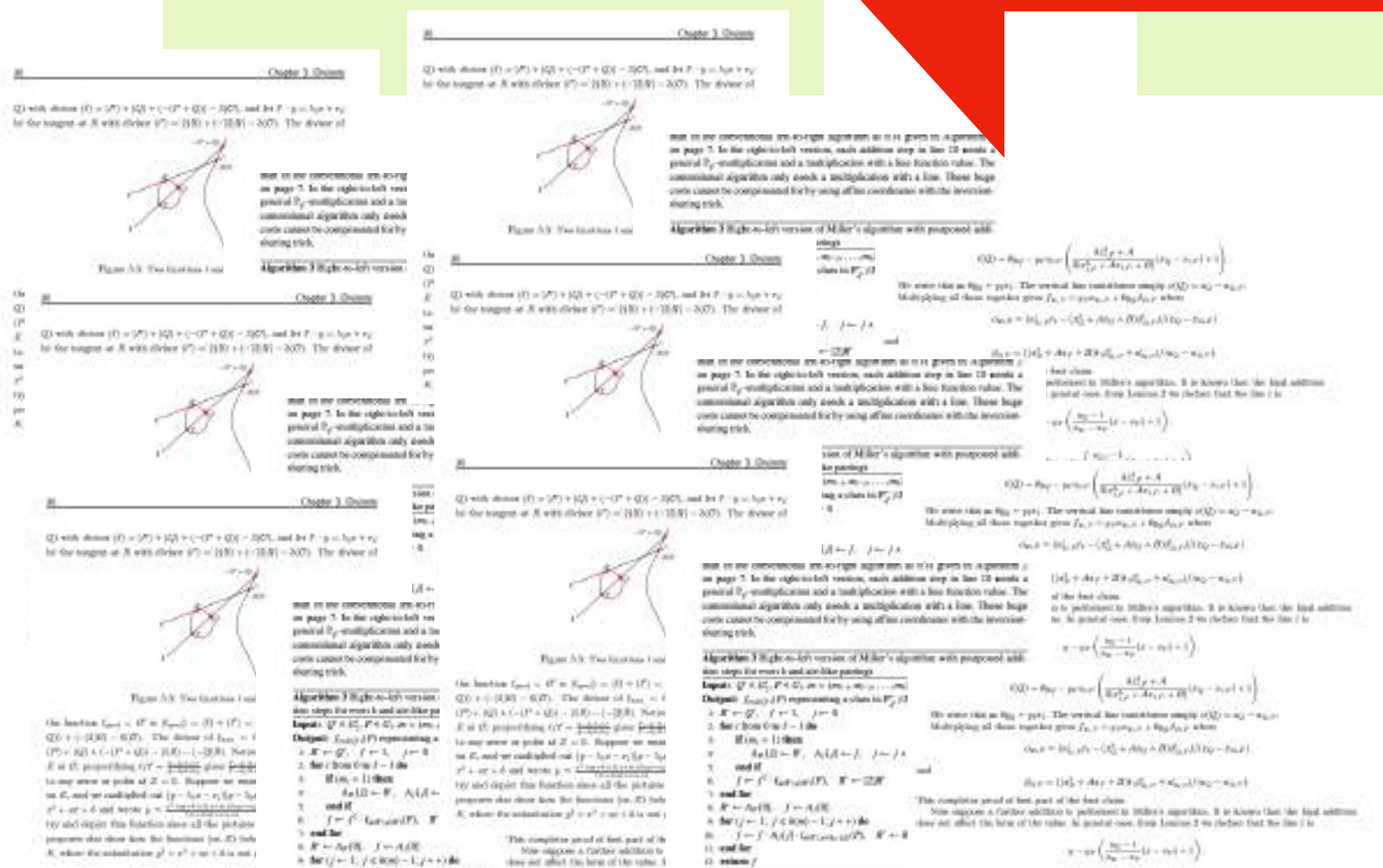
0 take some literature

1

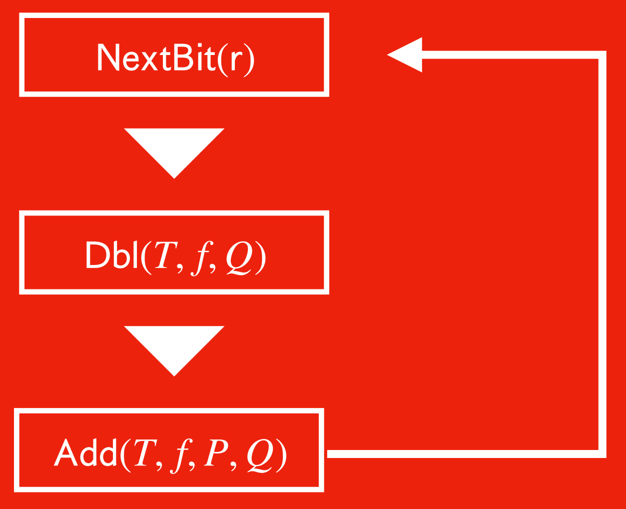
2



3 fast pairings



2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

general approach

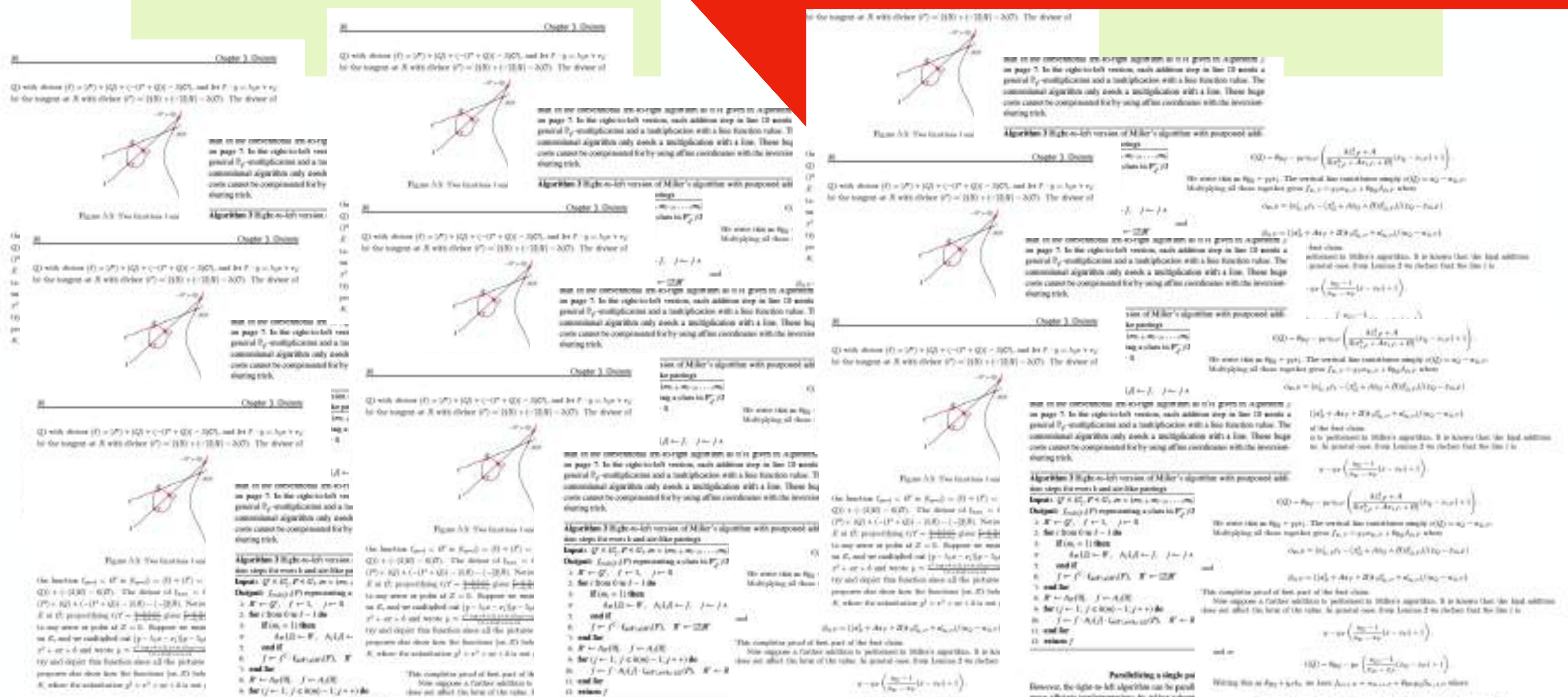
Instead I describe the general approach,
and leave all details out

✓

core idea

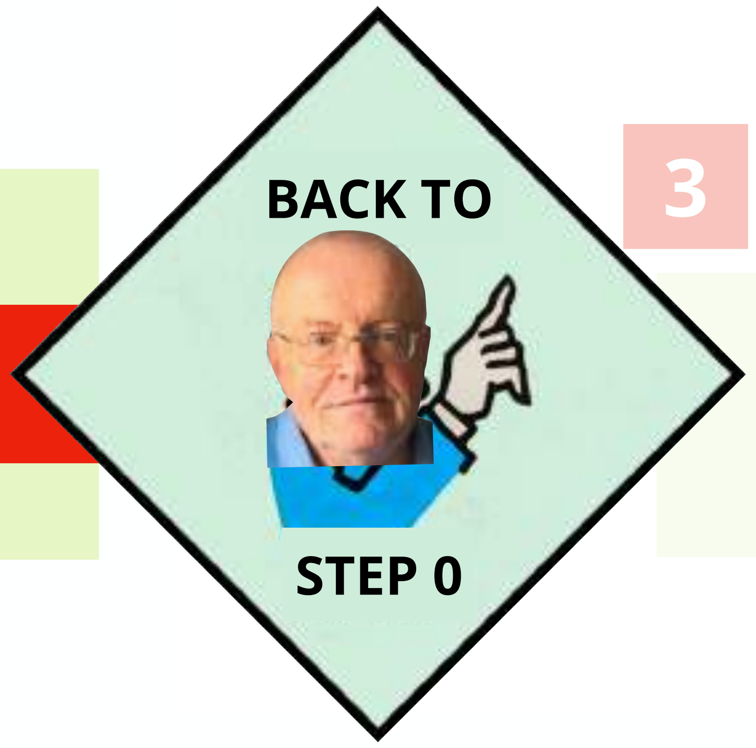
For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don't use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

0 take some literature



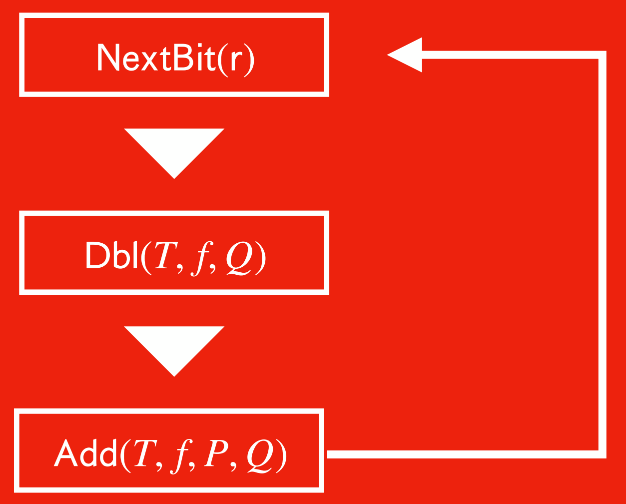
1

2



3 fast pairings

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

general approach

Instead I describe the general approach, and leave all details out

0 take some literature

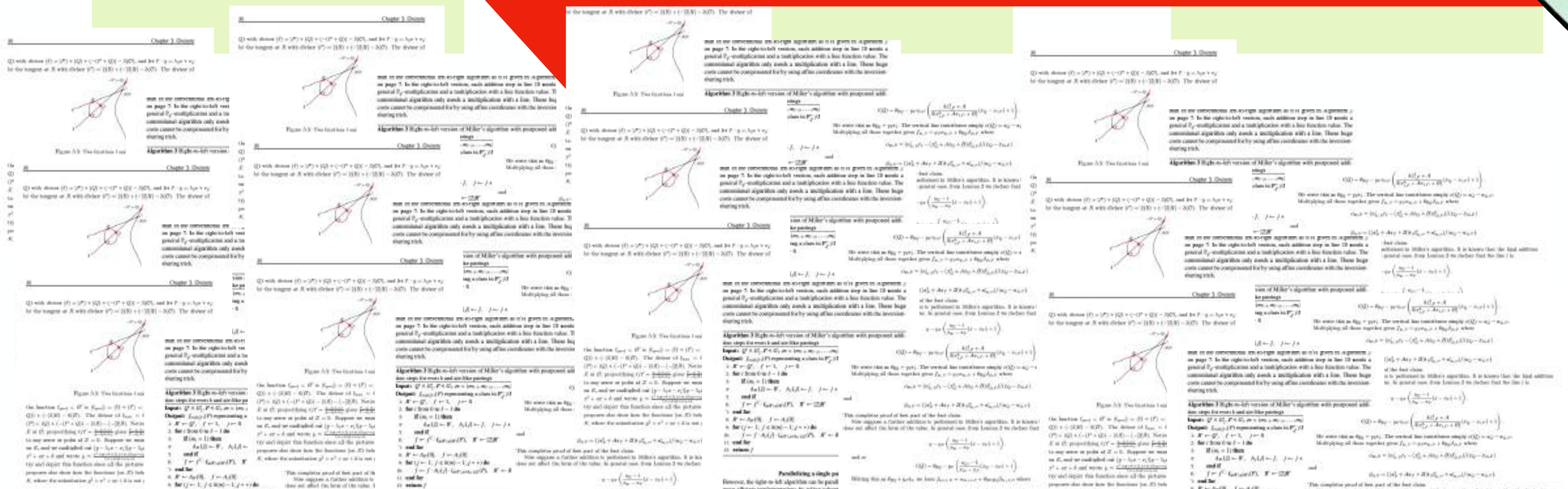
1

2

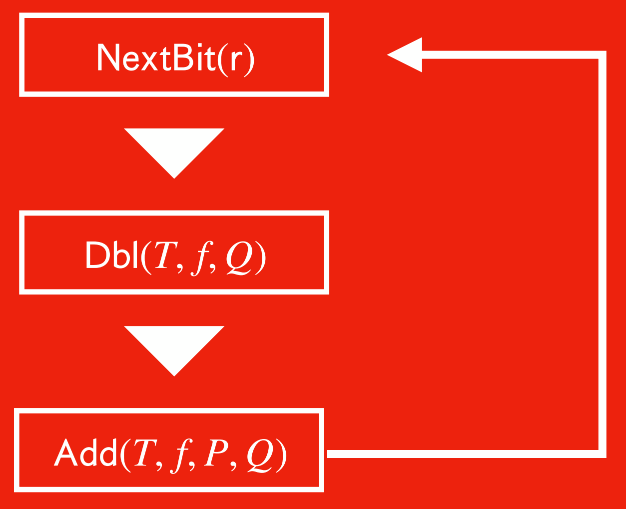
BACK TO

STEP 0

3 fast pairings



2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

general approach

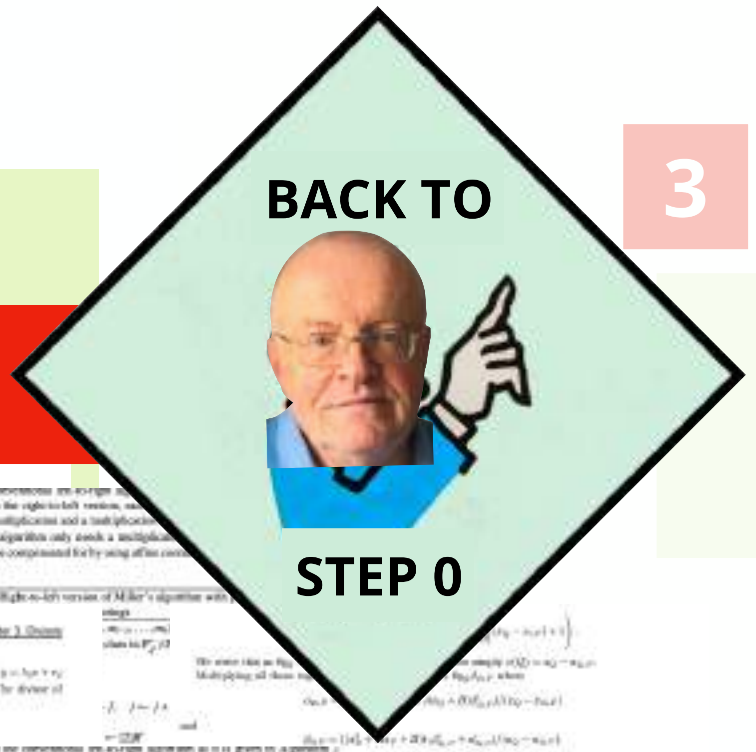
Instead I describe the general approach, and leave all details out

---

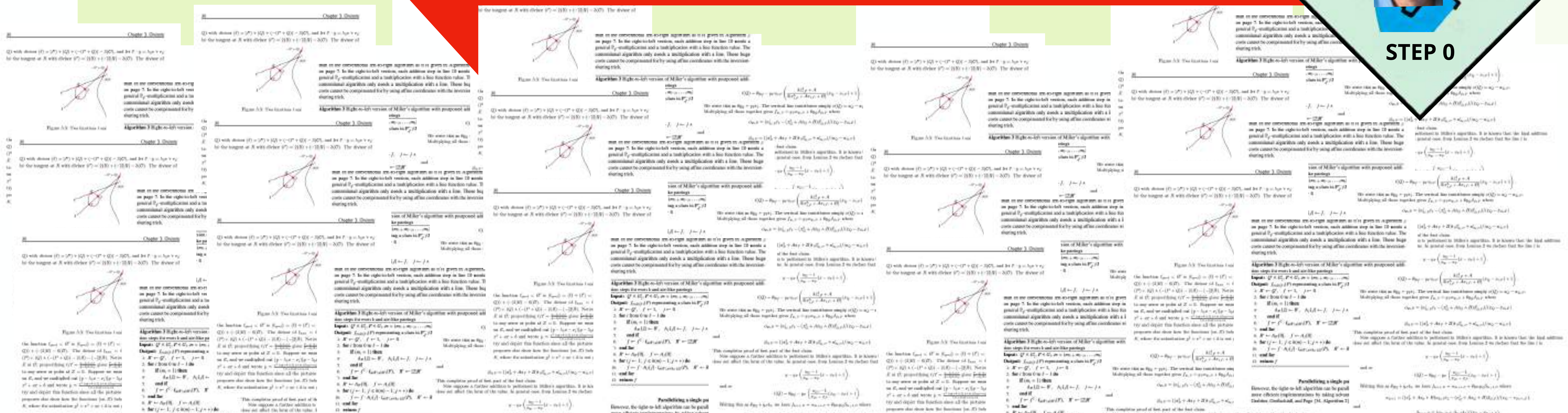
0 take some literature

1

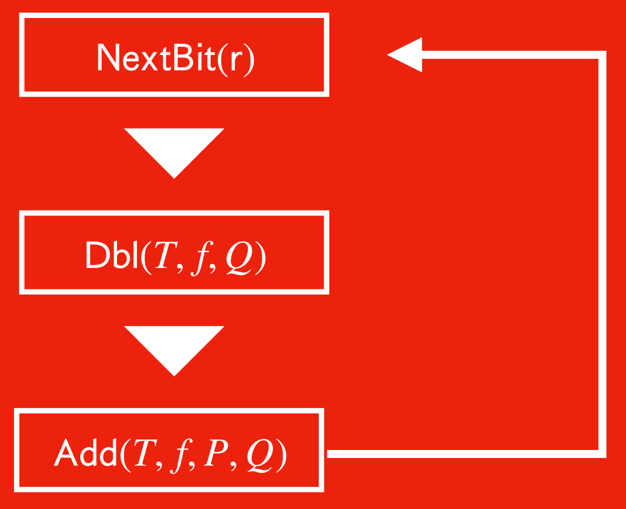
2



3 fast pairings



2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

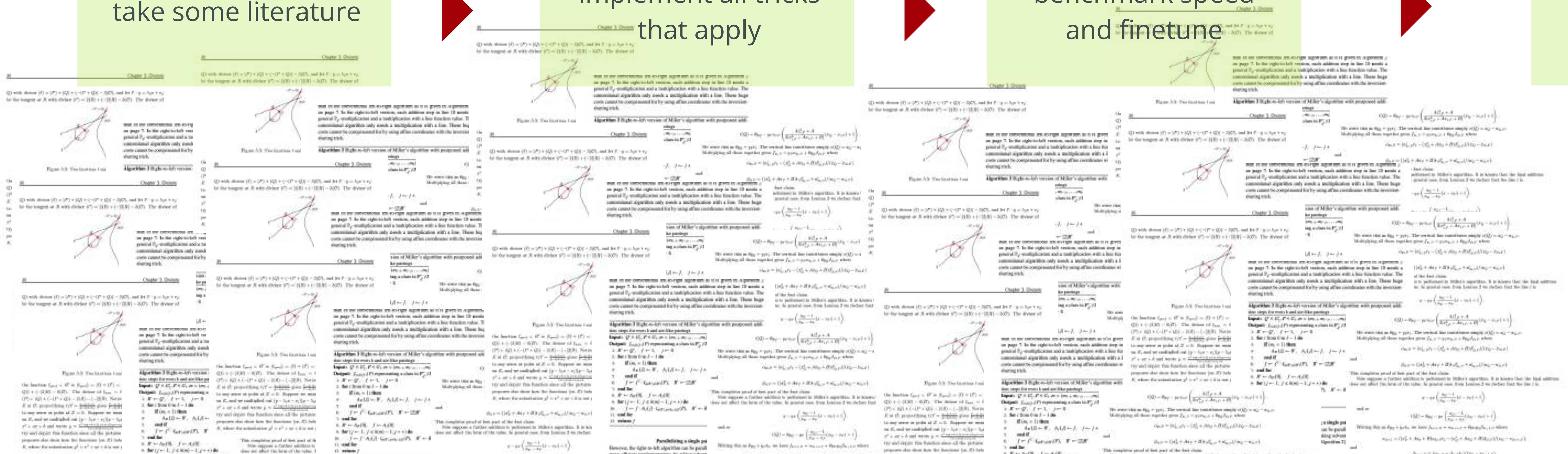
core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

✓

general approach

Instead I describe the general approach, and leave all details out



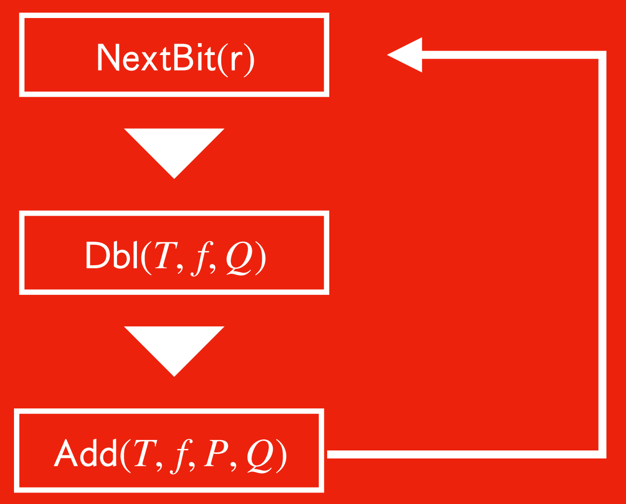
0 take some literature

1 implement all tricks that apply

2 benchmark speed and finetune

3 fast pairings

2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don’t use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

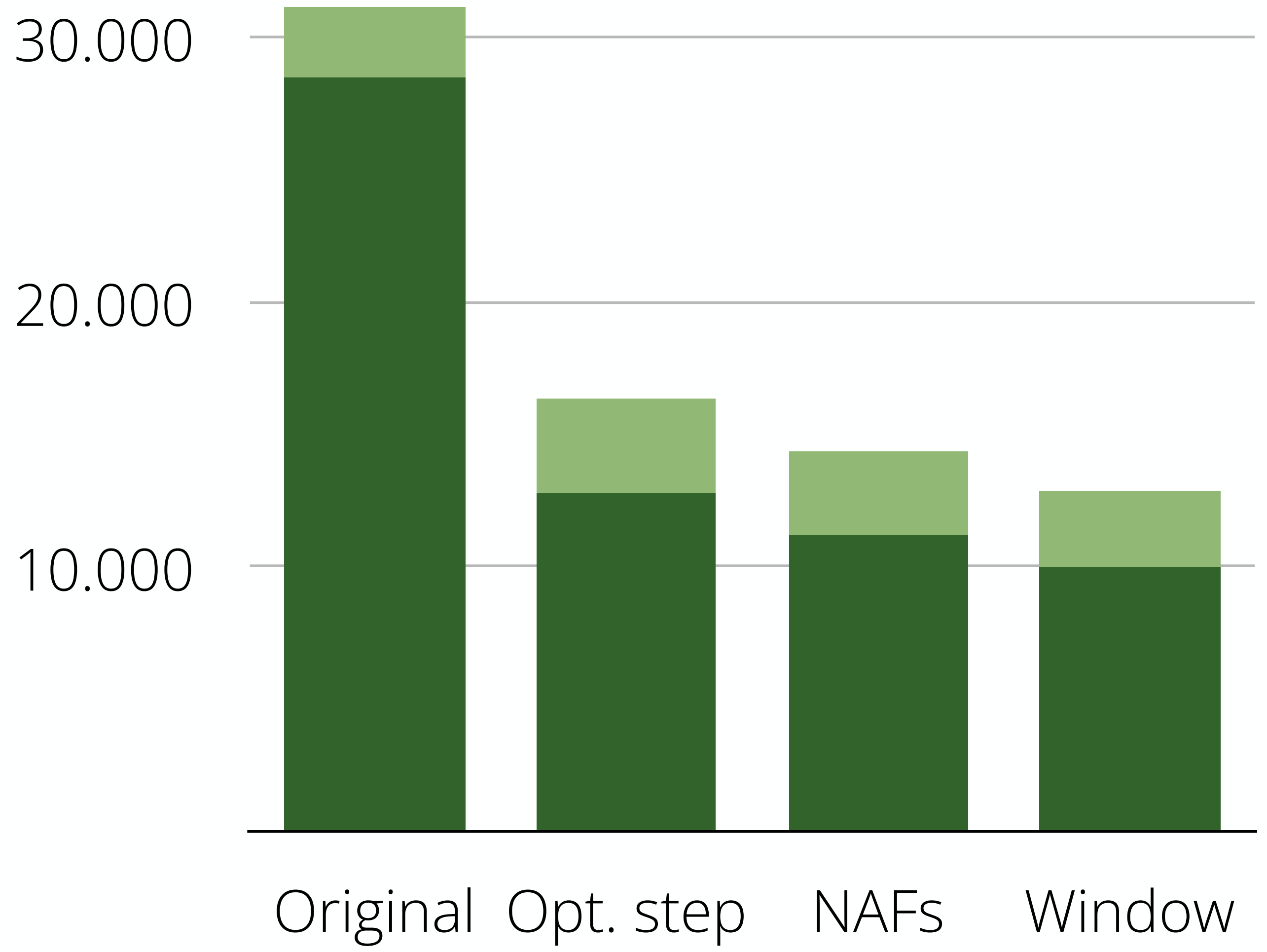
✓

general approach

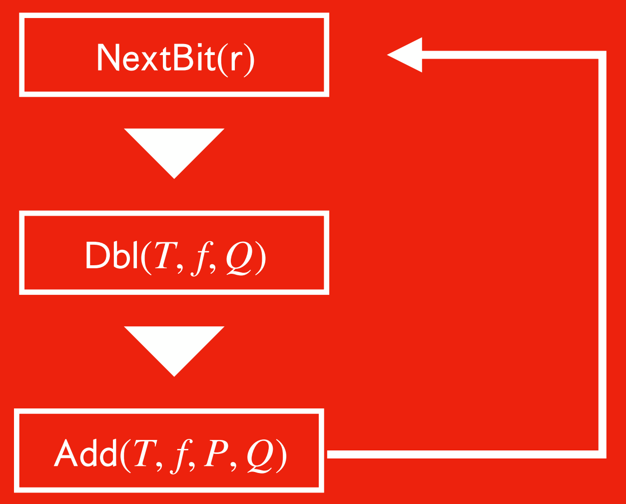
Instead I describe the general approach, and leave all details out

3

fast pairings



2



Speeding-up general pairings

!

general notice

Computing pairings fast is quite technical.
Better suited for papers than slides

✓

general approach

Instead I describe the general approach,
and leave all details out

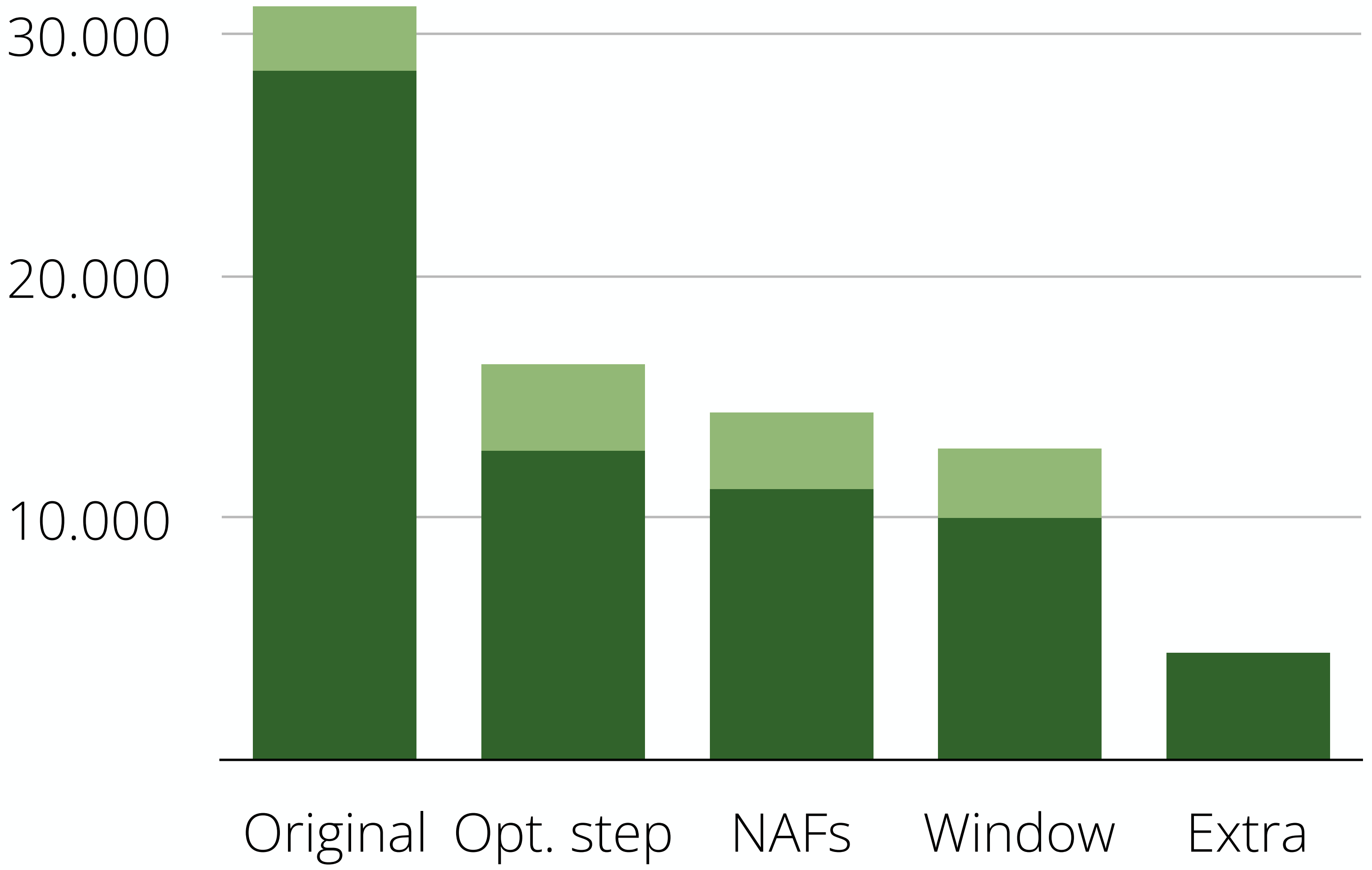
✓

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$,
don't use curve arithmetic
but pairing $e(P, Q)$ to get
overlap in orders!

3

fast pairings



extra pairings

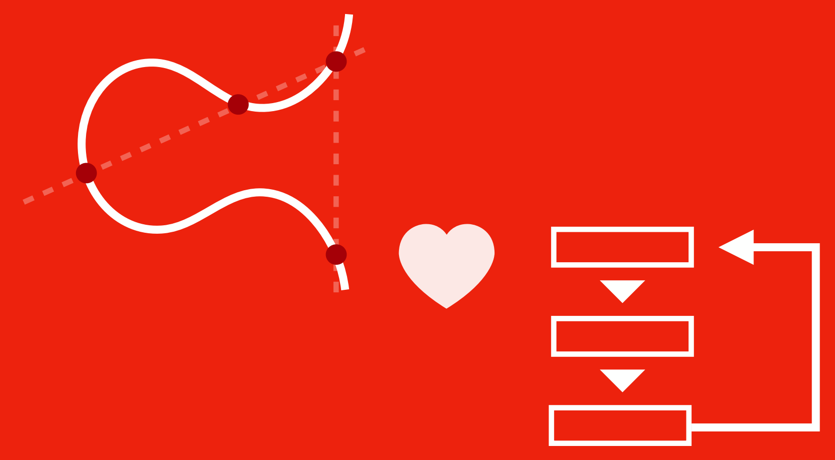
if you have already computed
$e(P, Q_1)$,

it is very efficient to compute
$e(P, Q_2)$

Fast pairings

❤

Isogeny crypto



Applying pairings in isogeny crypto

fast pairings

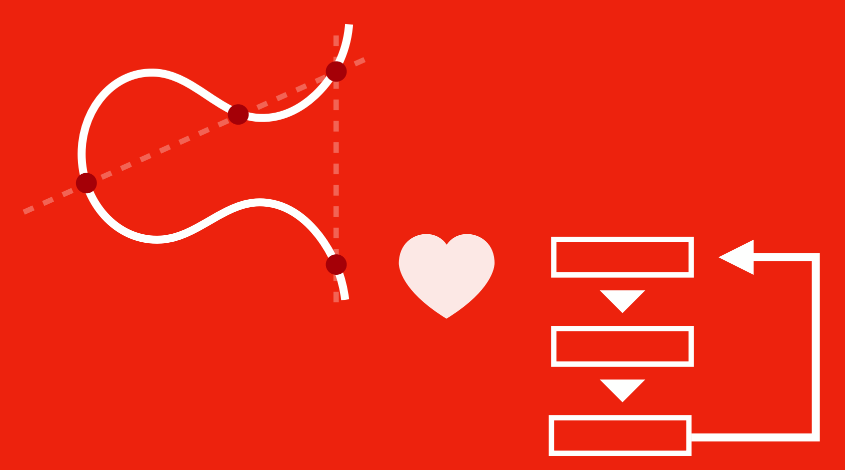
Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E'(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines



Applying pairings in isogeny crypto

✓ **fast pairings**

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E'(\mathbb{F}_p)$

&

✓ **core idea**

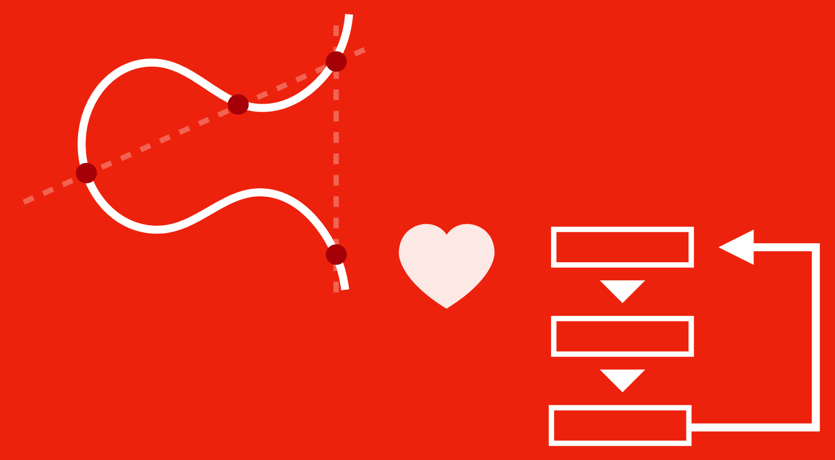
For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E'(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

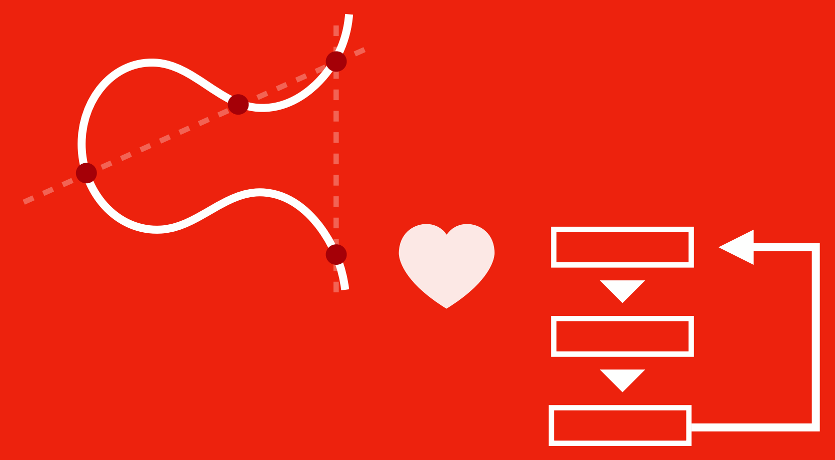
verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p + 1$, e.g. full torsion points

A: compute $\zeta = e(P, Q)$ and check that order ζ is $p + 1$.

3



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

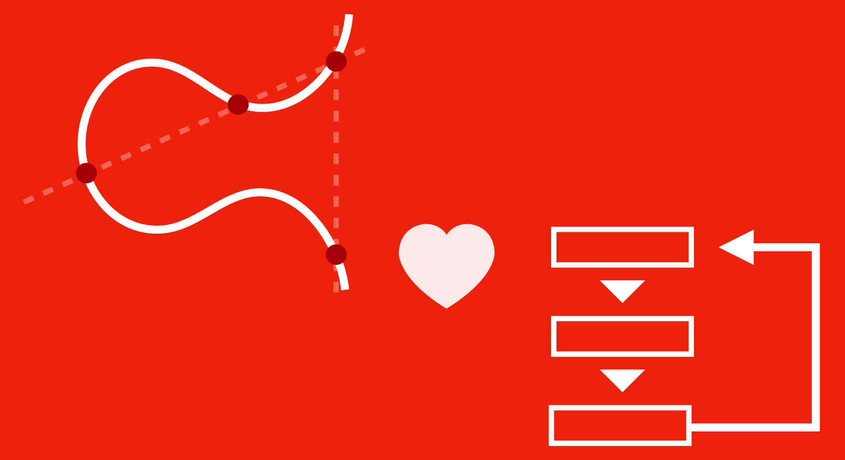
verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

A: compute $\zeta = e(P, Q)$ and check that order ζ is $p+1$.

speedup: -75%



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

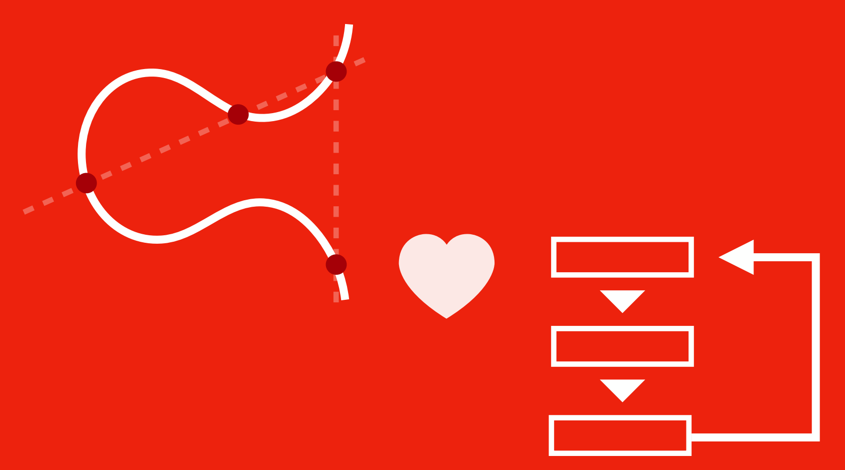
A: compute $\zeta = e(P, Q)$ and check that order ζ is $p+1$.

speedup: -75%

compute full torsion *P*

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$ of order $p+1$, e.g. full torsion points



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

A: compute $\zeta = e(P, Q)$ and check that order ζ is $p+1$.

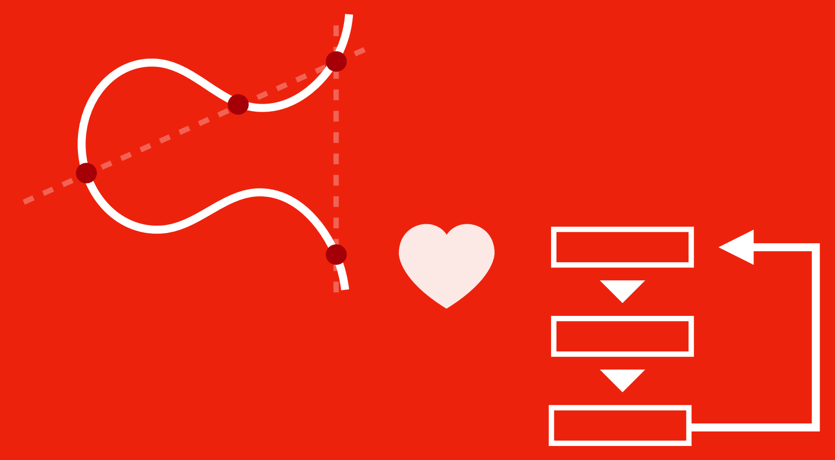
speedup: -75%

compute full torsion *P*

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$ of order $p+1$, e.g. full torsion points

A: take random, P, Q, then find $\zeta = e(P, Q)$. Compute order ζ and apply Gauss' algorithm.



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P,Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

A: compute $\zeta = e(P,Q)$ and check that order ζ is $p+1$.

speedup: -75%

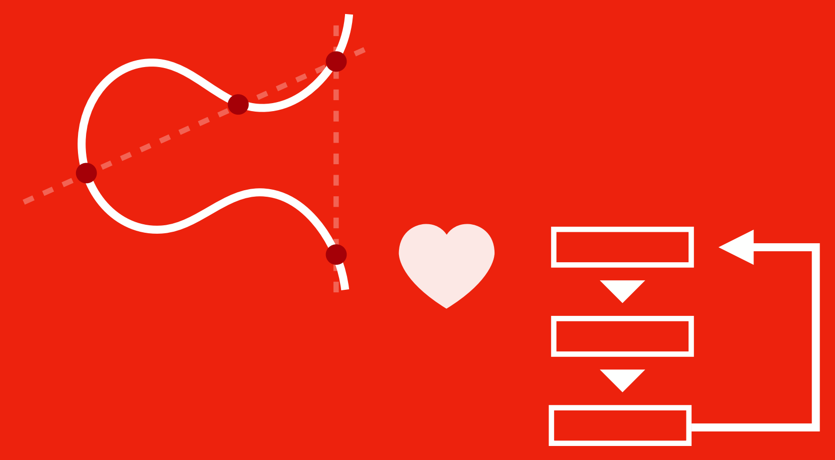
compute full torsion *P*

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$ of order $p+1$, e.g. full torsion points

A: take random, P, Q, then find $\zeta = e(P,Q)$. Compute order ζ and apply Gauss' algorithm.

speedup: case dependent, up to -75%



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E'(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion P

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

A: compute $\zeta = e(P, Q)$ and check that order ζ is $p+1$.

speedup: -75%

compute full torsion P

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E'(\mathbb{F}_p)$ of order $p+1$, e.g. full torsion points

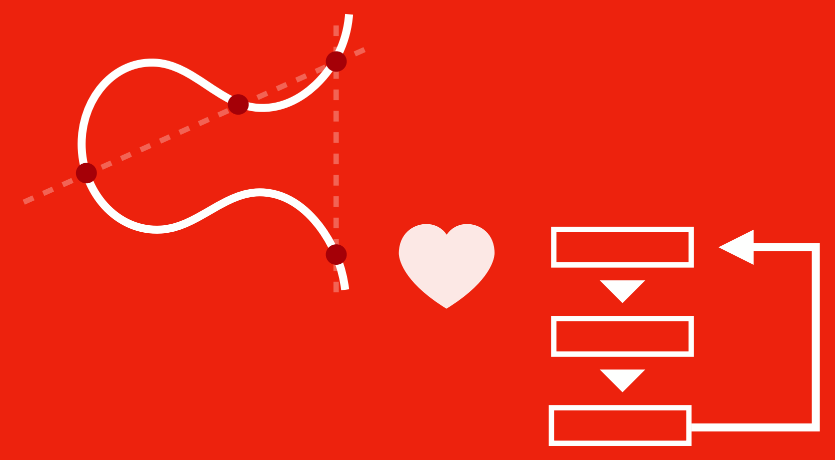
A: take random, P, Q, then find $\zeta = e(P, Q)$. Compute order ζ and apply Gauss' algorithm.

speedup: case dependent, up to -75%

verify supersingularity

In some CSIDH variants, we get E

Q: is E even supersingular? verify that it is!



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p + 1$, e.g. full torsion points

A: compute $\zeta = e(P, Q)$ and check that order ζ is $p + 1$.

speedup: -75%

compute full torsion *P*

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$ of order $p + 1$, e.g. full torsion points

A: take random, P, Q, then find $\zeta = e(P, Q)$. Compute order ζ and apply Gauss' algorithm.

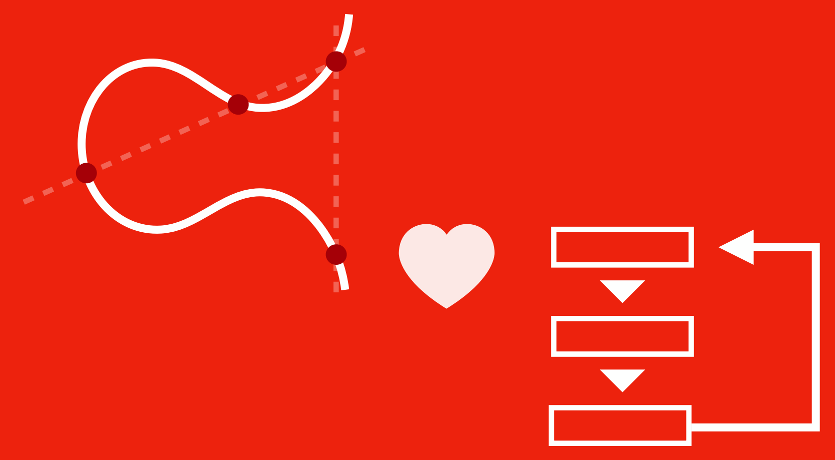
speedup: case dependent, up to -75%

verify supersingularity

In some CSIDH variants, we get E

Q: is E even supersingular? verify that it is!

A: take random, P, Q, then find $\zeta = e(P, Q)$. Verify order $\zeta \geq 4\sqrt{p}$.



Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P, Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

A: compute $\zeta = e(P, Q)$ and check that order ζ is $p+1$.

speedup: -75%

compute full torsion *P*

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$ of order $p+1$, e.g. full torsion points

A: take random, P, Q, then find $\zeta = e(P, Q)$. Compute order ζ and apply Gauss' algorithm.

speedup: case dependent, up to -75%

verify supersingularity

In some CSIDH variants, we get E

Q: is E even supersingular? verify that it is!

A: take random, P, Q, then find $\zeta = e(P, Q)$. Verify order $\zeta \geq 4\sqrt{p}$.

speedup: -27% compared to CSIDH's

Applying pairings in isogeny crypto

fast pairings

Optimized pairing computation for the specific scenario $P \in E(\mathbb{F}_p), Q \in E^t(\mathbb{F}_p)$

&

core idea

For $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$, don't use curve arithmetic but pairing $e(P,Q)$ to get overlap in orders!

Faster isogeny subroutines

verify full torsion *P*

In some CSIDH variants, we are given $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$.

Q: verify that both P and Q have order $p+1$, e.g. full torsion points

A: compute $\zeta = e(P,Q)$ and check that order ζ is $p+1$.

speedup: -75%

compute full torsion *P*

In some CSIDH variants, we get E

Q: find $P \in E(\mathbb{F}_p)$ and $Q \in E^t(\mathbb{F}_p)$ of order $p+1$, e.g. full torsion points

A: take random, P, Q, then find $\zeta = e(P,Q)$. Compute order ζ and apply Gauss' algorithm.

speedup: case dependent, up to -75%

verify supersingularity

In some CSIDH variants, we get E

Q: is E even supersingular? verify that it is!

A: take random, P, Q, then find $\zeta = e(P,Q)$. Verify order $\zeta \geq 4\sqrt{p}$.

speedup: +2% compared to Doliskani

?

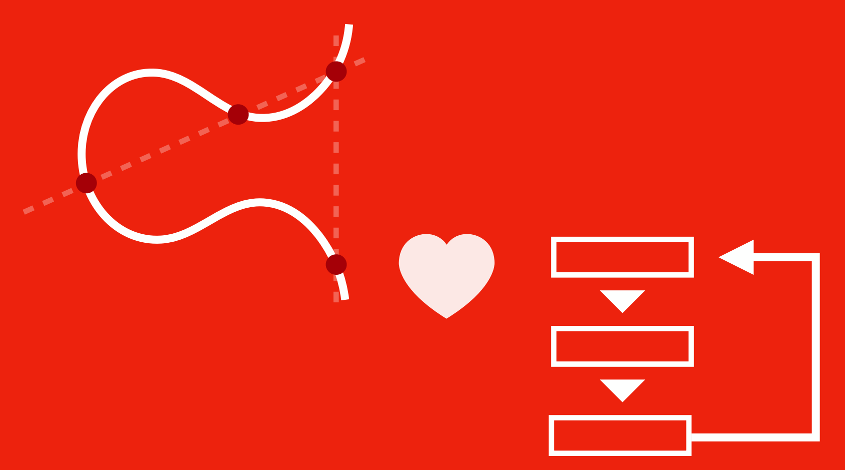
3

Applying pairings in isogeny crypto

why pairings at all?

scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free



Applying pairings in isogeny crypto

why pairings at all?

scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free

CSIDH's maturity?

3

Applying pairings in isogeny crypto

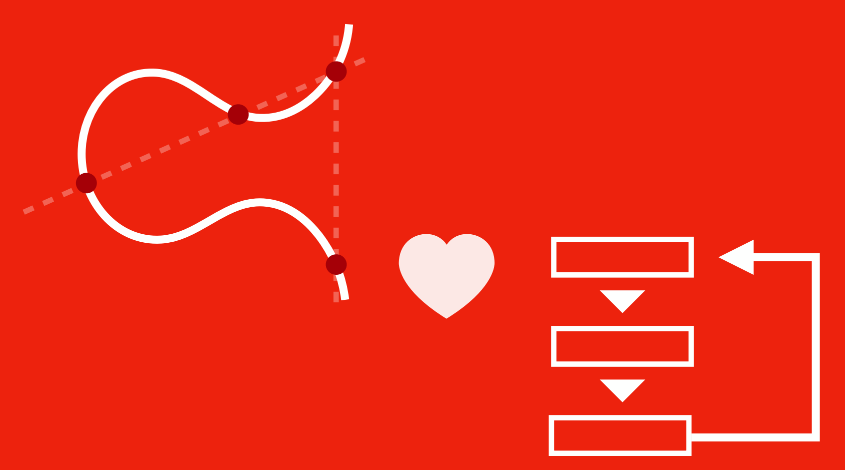
why pairings at all?

scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free

CSIDH's maturity?

✓ classical security well understood



Applying pairings in isogeny crypto

why pairings at all?

scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free

CSIDH's maturity?

- ✓ classical security well understood
- ? quantum security well understood

Applying pairings in isogeny crypto

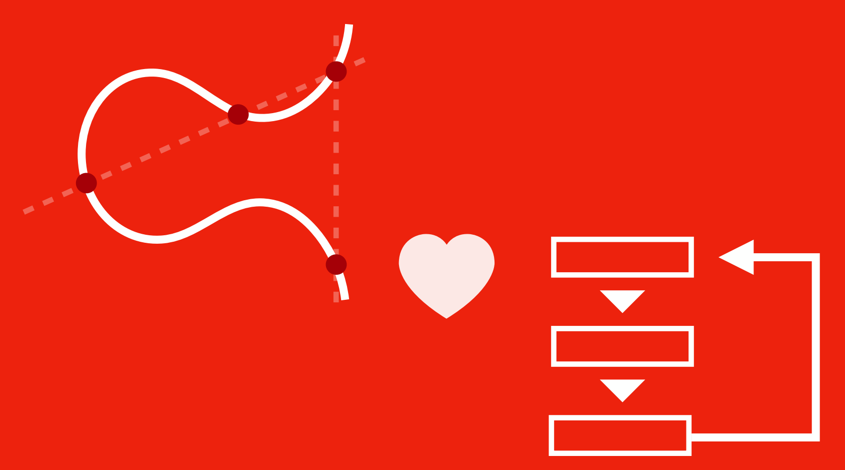
why pairings at all?

scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free

CSIDH's maturity?

- ✓ classical security well understood
- ? quantum security well understood
- ? quite slow constant-time



Applying pairings in isogeny crypto

why pairings at all?

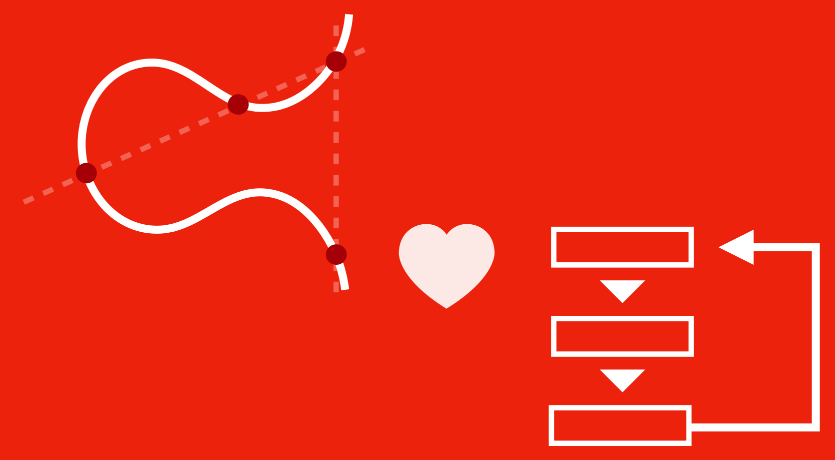
scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free

CSIDH's maturity?

- ✓ classical security well understood
- ? quantum security well understood
- ? quite slow constant-time
- ✗ *very slow* deterministic, dummy-free

3



Applying pairings in isogeny crypto

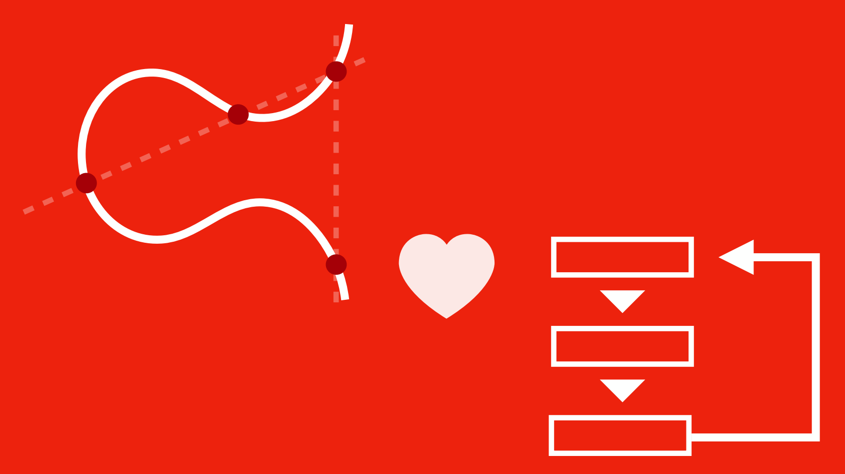
why pairings at all?

scheme maturity	CSIDH's maturity?
classical security well understood	✓ classical security well understood
quantum security well understood	? quantum security well understood
fast, constant-time implementation	? quite slow constant-time
deterministic and dummy-free	✗ *very slow* deterministic, dummy-free

how do we achieve fast high-security CSIDH?

constant-time, deterministic, dummy-free

✗ **previously** → ? **with pairings** → ✓ ***to do***



Applying pairings in isogeny crypto

why pairings at all?

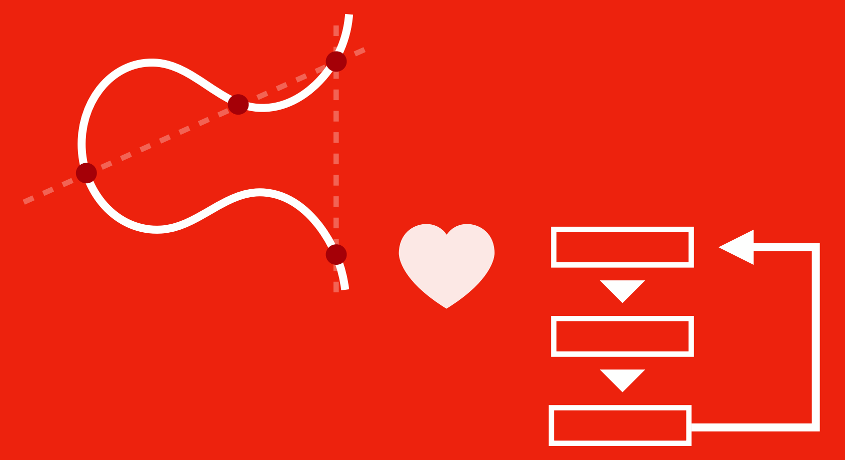
scheme maturity	CSIDH's maturity?
classical security well understood	✓ classical security well understood
quantum security well understood	? quantum security well understood
fast, constant-time implementation	? quite slow constant-time
deterministic and dummy-free	✗ *very slow* deterministic, dummy-free

how do we achieve fast high-security CSIDH?

constant-time, deterministic, dummy-free

✗ **previously** → ? **with pairings** → ✓ ***to do***

- add **seed** for torsion points in key
- **slow** verification of torsion points
- **slow** group action due to dummy-free



Applying pairings in isogeny crypto

why pairings at all?

scheme maturity

- classical security well understood
- quantum security well understood
- fast, constant-time implementation
- deterministic and dummy-free

CSIDH's maturity?

- ✓ classical security well understood
- ? quantum security well understood
- ? quite slow constant-time
- ✗ *very slow* deterministic, dummy-free

how do we achieve fast high-security CSIDH?

constant-time, deterministic, dummy-free

✗ → ? → ✓

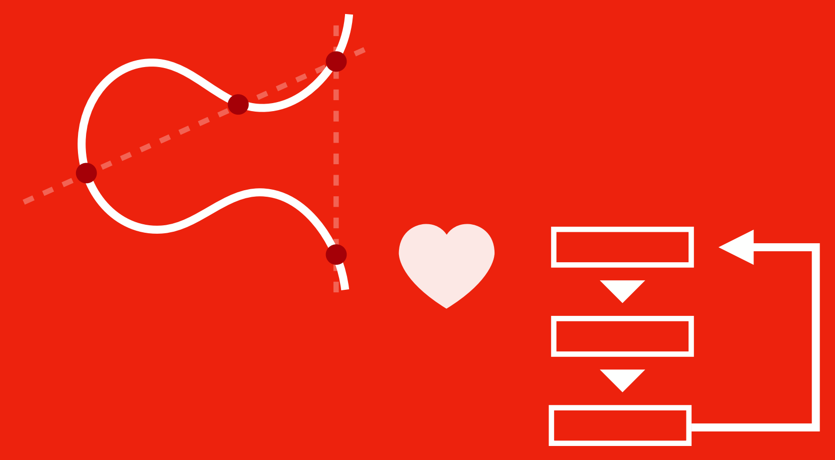
previously

- add **seed** for torsion points in key
- **slow** verification of torsion points
- **slow** group action due to dummy-free

with pairings

- **fast** verification of torsion points
- removes probability from CTIDH
- improved group action and ss verify!

to do



Applying pairings in isogeny crypto

why pairings at all?

scheme maturity	**CSIDH's maturity?**
classical security well understood	✓ classical security well understood
quantum security well understood	? quantum security well understood
fast, constant-time implementation	? quite slow constant-time
deterministic and dummy-free	✗ *very slow* deterministic, dummy-free

how do we achieve fast high-security CSIDH?

constant-time, deterministic, dummy-free

✗ previously

- add **seed** for torsion points in key
- **slow** verification of torsion points
- **slow** group action due to dummy-free

? with pairings

- **fast** verification of torsion points
- removes probability from CTIDH
- improved group action and ss verify!

✓ *to do*

- analyse **optimal** use of torsion
- can we use **faster** torsion finding?
- can improve group action!

Thank you!

Any questions*?

*If not, I have a question for you...

Constant-time Gauss' algorithm?

Finite field world

Q: Given $\mathbb{F}_q$ find generator ζ for $\mathbb{F}_q^*$

Curve world

Given curve E over $\mathbb{F}_p$,
find full torsion point P

Constant-time Gauss' algorithm?

Finite field world

Q: Given $\mathbb{F}_q$ find generator ζ for $\mathbb{F}_q^*$

A:

GAUSS' ALGORITHM

1. Take random $\zeta \in \mathbb{F}_q$, compute $t = \text{Order}(\,\zeta\,)$
2. If $t = q - 1$, **stop**,
3. **else** take random $\beta \in \mathbb{F}_q^*$ and compute $s = \text{Order}(\,\beta\,)$
   a. if $s = q - 1$, **stop**
   b. **else** find coprime $d \mid t$ and $e \mid s$ with $d \cdot e = \text{lcm}(t, s)$
   c. set $\zeta \leftarrow \zeta^{t/d} \cdot \beta^{s/e}$ and $t \leftarrow d \cdot e$ and **repeat** from 2.

Curve world

Given curve E over $\mathbb{F}_p$,
find full torsion point P

Take P and Q,
Compute their torsion.
If P not full torsion,
take right multiple Q
set $P \leftarrow P + Q$ to fill
missing torsion in P
repeat until full torsion

Constant-time Gauss' algorithm?

Finite field world

Q: Given $\mathbb{F}_q$ find generator ζ for $\mathbb{F}_q^*$

A:

Gauss' Algorithm

1. Take random $\zeta \in \mathbb{F}_q$, compute $t = \mathrm{Order}(\zeta)$
2. If $t = q - 1$, **stop**,
3. **else** take random $\beta \in \mathbb{F}_q^*$ and compute $s = \mathrm{Order}(\beta)$
   a. if $s = q - 1$, **stop**
   b. **else** find coprime $d \mid t$ and $e \mid s$ with $d \cdot e = \mathrm{lcm}(t, s)$
   c. set $\zeta \leftarrow \zeta^{t/d} \cdot \beta^{s/e}$ and $t \leftarrow d \cdot e$ and **repeat** from 2.

Q: Given $\mathbb{F}_q$ find generator ζ for $\mathbb{F}_q^*$ *in constant-time*

Curve world

Given curve E over $\mathbb{F}_p$,
find full torsion point P

Take P and Q,
Compute their torsion.
If P not full torsion,
take right multiple Q
set $P \leftarrow P + Q$ to fill
missing torsion in P
repeat until full torsion

Given curve E over $\mathbb{F}_p$,
find full torsion point P
in constant-time