CT-RSA 2012 — February 29th, 2012

Optimal Eta Pairing on Supersingular Genus-2 Binary Hyperelliptic Curves

Nicolas Estibals

CARAMEL project-team, LORIA,
Université de Lorraine / CNRS / INRIA, France
Nicolas.Estibals@loria.fr

Joint work with:

Diego F. Aranha	Institute of Computing, University of Campinas, Brazil
Jean-Luc Beuchat	Graduate School of Systems and Information Engineering, University of Tsukuba, Japan
Jérémie Detrey	CARAMEL project-team, LORIA, INRIA / Université de Lorraine / CNRS, France

Pairings and cryptology

- used as a primitive in many protocols and devices
  - Boneh–Lynn–Shacham short signature
  - Boneh–Franklin identity-based encryption
  - ...

Pairings and cryptology

- used as a primitive in many protocols and devices
  - Boneh–Lynn–Shacham short signature
  - Boneh–Franklin identity-based encryption
  - ...
- implementations needed for various targets
  - online server $\rightarrow$ high-speed software
  - smart card $\rightarrow$ low-resource hardware
- reach 128 bits of security (equivalent to AES)

What's a cryptographic pairing

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$

- where $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \times)$ are cyclic groups of order ℓ
- The discrete logarithm problem should be hard on these groups

What's a cryptographic pairing

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$

- where $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \times)$ are cyclic groups of order ℓ
- The discrete logarithm problem should be hard on these groups
- Bilinear map:
$$e(aP, bQ) = e(P, Q)^{ab}$$

What's a cryptographic pairing

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$

- where $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \times)$ are cyclic groups of order ℓ
- The discrete logarithm problem should be hard on these groups
- Bilinear map:
$$e(aP, bQ) = e(P, Q)^{ab}$$
- Symmetric pairing (Type-1): $\mathbb{G}_1 = \mathbb{G}_2$, exploited by some protocols

What's a cryptographic pairing

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$

- where $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \times)$ are cyclic groups of order ℓ
- The discrete logarithm problem should be hard on these groups
- Bilinear map:
$$e(aP, bQ) = e(P, Q)^{ab}$$
- Symmetric pairing (Type-1): $\mathbb{G}_1 = \mathbb{G}_2$, exploited by some protocols
- Choice of the groups:
  - $\mathbb{G}_1$, $\mathbb{G}_2$: related to an algebraic curve
  - $\mathbb{G}_T$: related to the field of definition of the curve

Classical choice of curves

Barreto–Naehrig curves

- \+ Lots of literature
- \+ Huge optimization efforts
- \+ Suited for 128 bits of security
- – Arithmetic modulo $p \approx 256$ bits
- – No symmetric pairing

Classical choice of curves

Barreto–Naehrig curves

+ Lots of literature
+ Huge optimization efforts
+ Suited for 128 bits of security
– Arithmetic modulo $p \approx 256$ bits
– No symmetric pairing

Supersingular elliptic curves

+ Symmetric pairing
  Thanks to a distortion map
  $\psi : \mathbb{G}_1 \to \mathbb{G}_2$
+ Small characteristic arithmetic
  $\Rightarrow$ No carry propagation
– Not suited to 128-bit security level
  Larger base field: $\mathbb{F}_{2^{1223}}$, $\mathbb{F}_{3^{509}}$

Classical choice of curves

Barreto–Naehrig curves

+ Lots of literature
+ Huge optimization efforts
+ Suited for 128 bits of security
– Arithmetic modulo $p \approx 256$ bits
– No symmetric pairing

Supersingular elliptic curves

+ Symmetric pairing

  Thanks to a distortion map

  $\psi : \mathbb{G}_1 \to \mathbb{G}_2$

+ Small characteristic arithmetic

  $\Rightarrow$ No carry propagation

– Not suited to 128-bit security level

  Larger base field: $\mathbb{F}_{2^{1223}}$, $\mathbb{F}_{3^{509}}$

Classical choice of curves

Barreto–Naehrig curves

+ Lots of literature
+ Huge optimization efforts
+ Suited for 128 bits of security
– Arithmetic modulo $p \approx 256$ bits
– No symmetric pairing

Supersingular elliptic curves

+ Symmetric pairing
  Thanks to a distortion map
  $\psi : \mathbb{G}_1 \rightarrow \mathbb{G}_2$
+ Small characteristic arithmetic
  $\Rightarrow$ No carry propagation
– Not suited to 128-bit security level
  Larger base field: $\mathbb{F}_{2^{1223}}$, $\mathbb{F}_{3^{509}}$

- Solutions to the large base field needed by supersingular curves
  - (Pairing 2010) Use fields of composite extension degree: benefit from faster field arithmetic but requires careful security analysis

Classical choice of curves

Barreto–Naehrig curves

+ Lots of literature
+ Huge optimization efforts
+ Suited for 128 bits of security
– Arithmetic modulo $p \approx 256$ bits
– No symmetric pairing

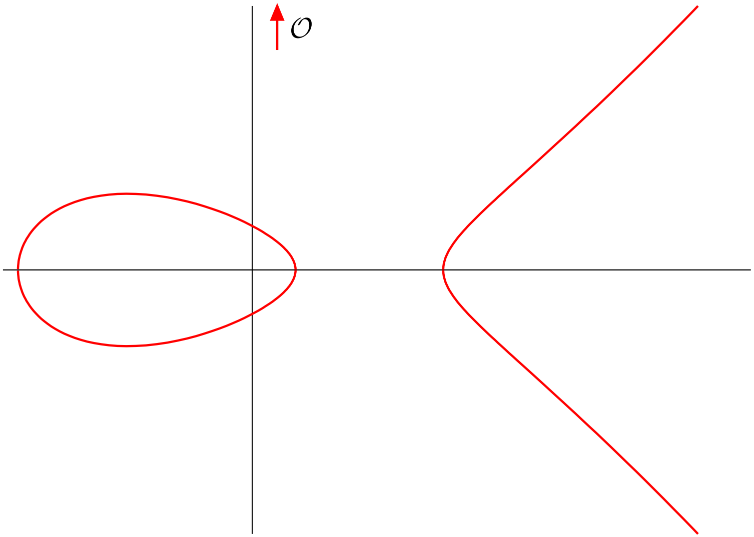
Supersingular elliptic curves

+ Symmetric pairing
  Thanks to a distortion map
  $\psi : \mathbb{G}_1 \rightarrow \mathbb{G}_2$
+ Small characteristic arithmetic
  $\Rightarrow$ No carry propagation
– Not suited to 128-bit security level
  Larger base field: $\mathbb{F}_{2^{1223}}$, $\mathbb{F}_{3^{509}}$

- Solutions to the large base field needed by supersingular curves
  - (Pairing 2010) Use fields of composite extension degree: benefit from faster field arithmetic but requires careful security analysis
  - (This work) Use genus-2 hyperelliptic curves: base field will be $\mathbb{F}_{2^{367}}$

Elliptic curves

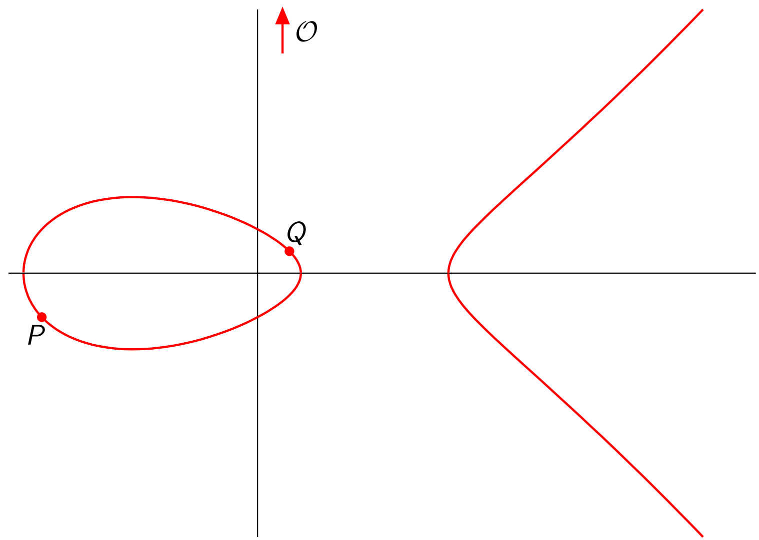
$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Elliptic curves

- $E(K)$ is a group

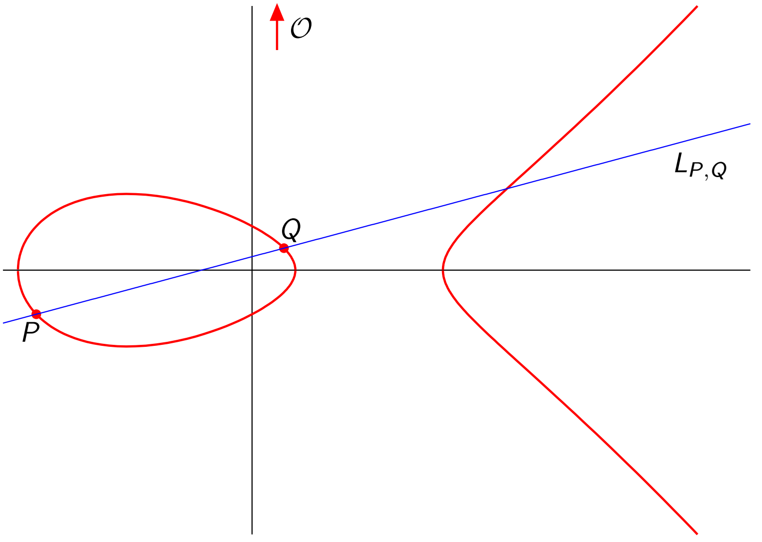
$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Elliptic curves

- $E(K)$ is a group

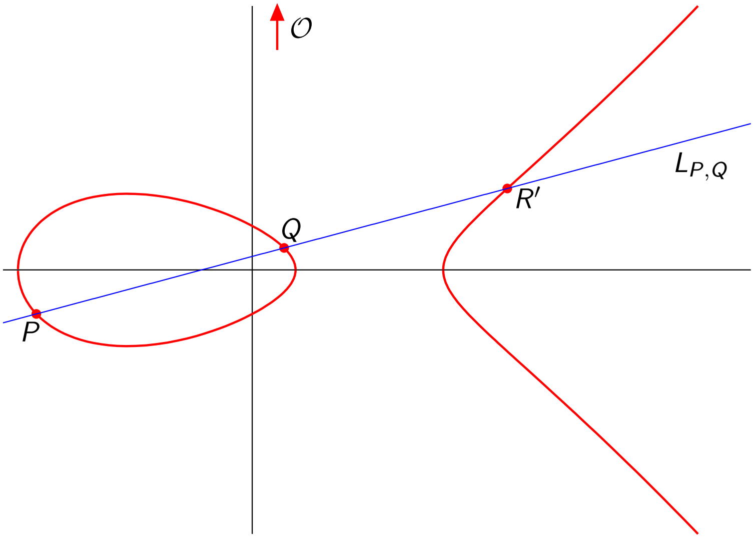
$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Elliptic curves

- $E(K)$ is a group

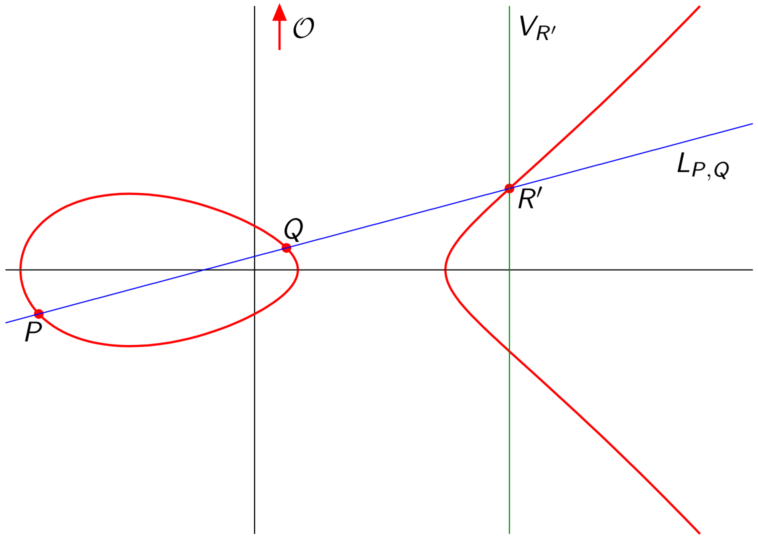
$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Elliptic curves

- $E(K)$ is a group

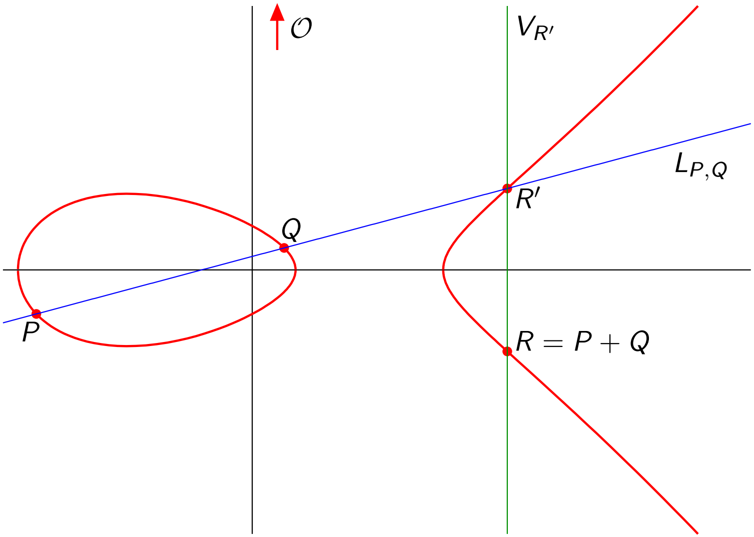
$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Elliptic curves

- $E(K)$ is a group

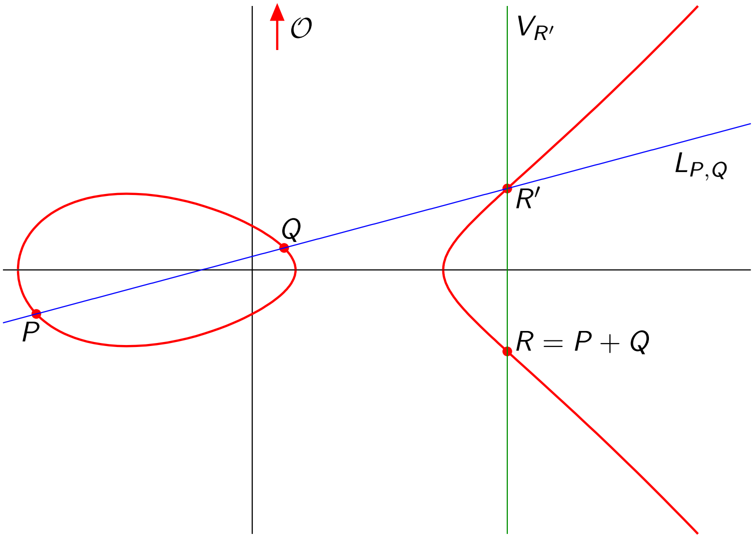
$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Elliptic curves

- $E(K)$ is a group
- In practice: K is a finite field $\mathbb{F}_q$
- $E(\mathbb{F}_q)$ is a finite group

$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$

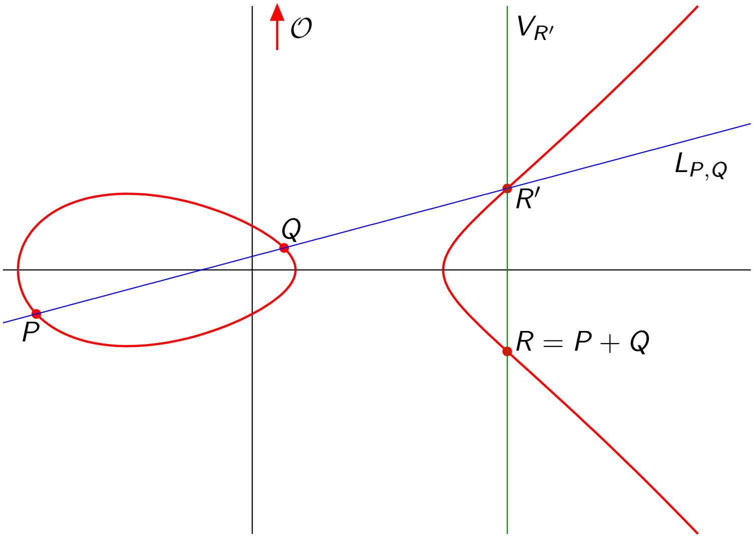


Elliptic curves

- $E(K)$ is a group
- In practice: K is a finite field $\mathbb{F}_q$
- $E(\mathbb{F}_q)$ is a finite group
- ℓ: a large prime dividing $\#E(\mathbb{F}_q)$
- Use the cyclic subgroup

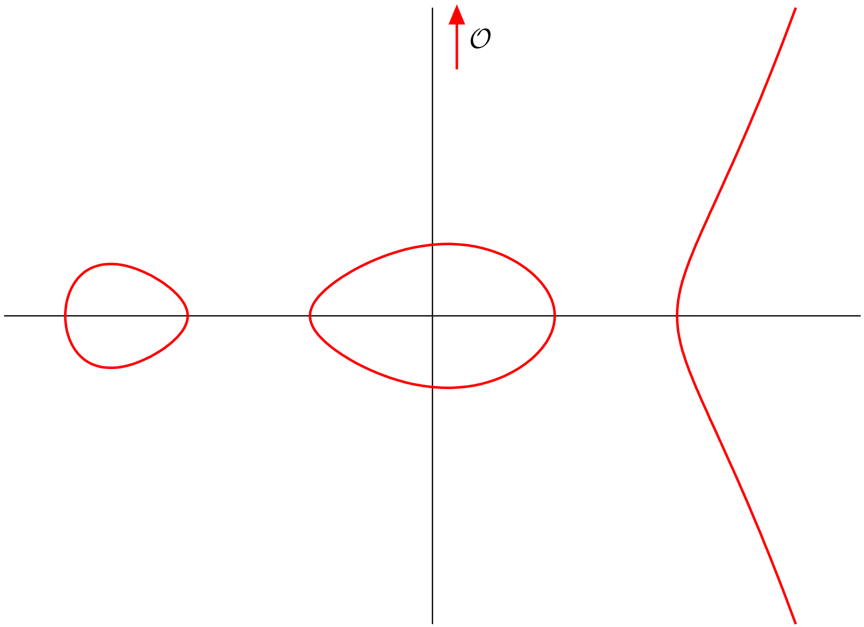
$$E(\mathbb{F}_q)[\ell] = \{P \mid [\ell]P = \mathcal{O}\}$$

$$E/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



Genus-2 hyperelliptic curves

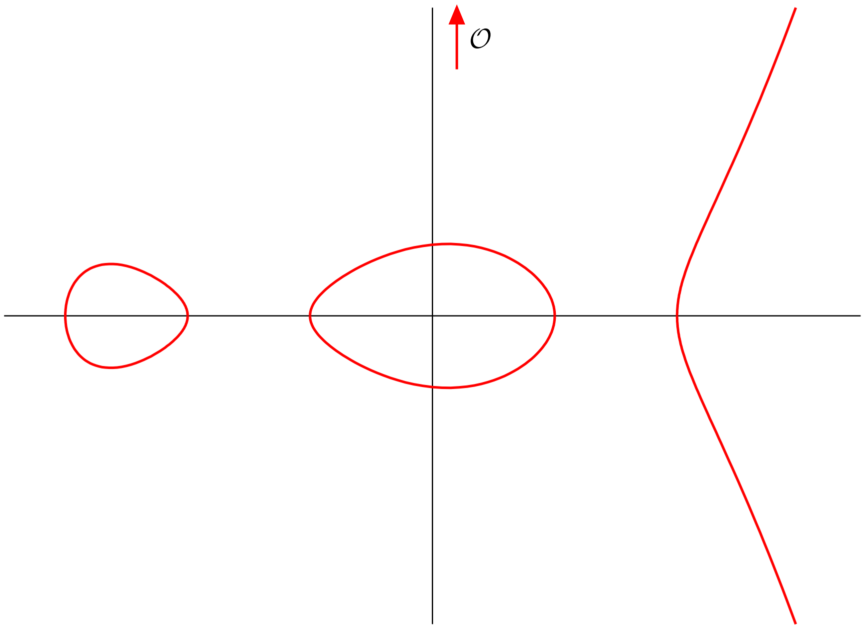
$$C/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

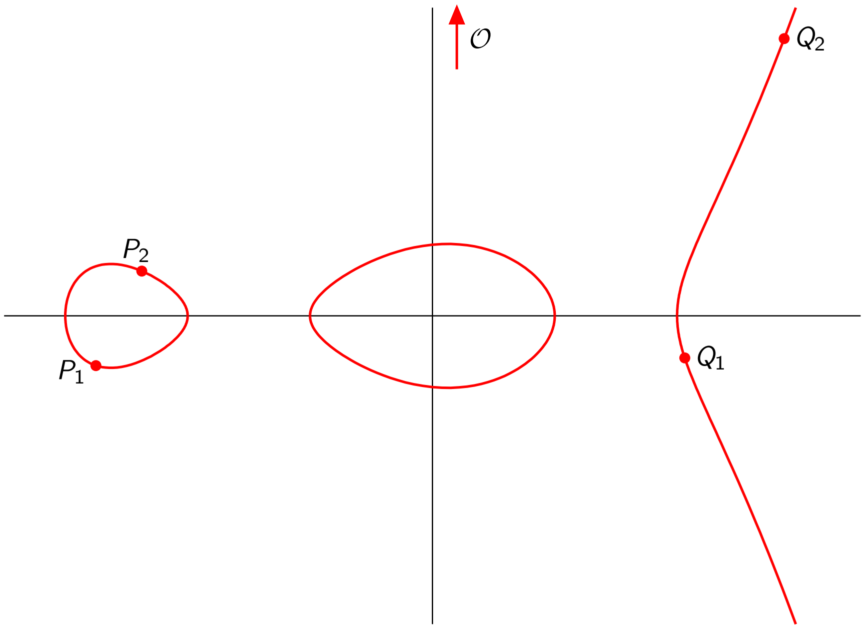
- $C(K)$ not a group!



Genus-2 hyperelliptic curves

- $C(K)$ not a group!
- But pairs of points
  $\{P_1, P_2\}$

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with deg $h \leq 2$ and deg $f = 5$

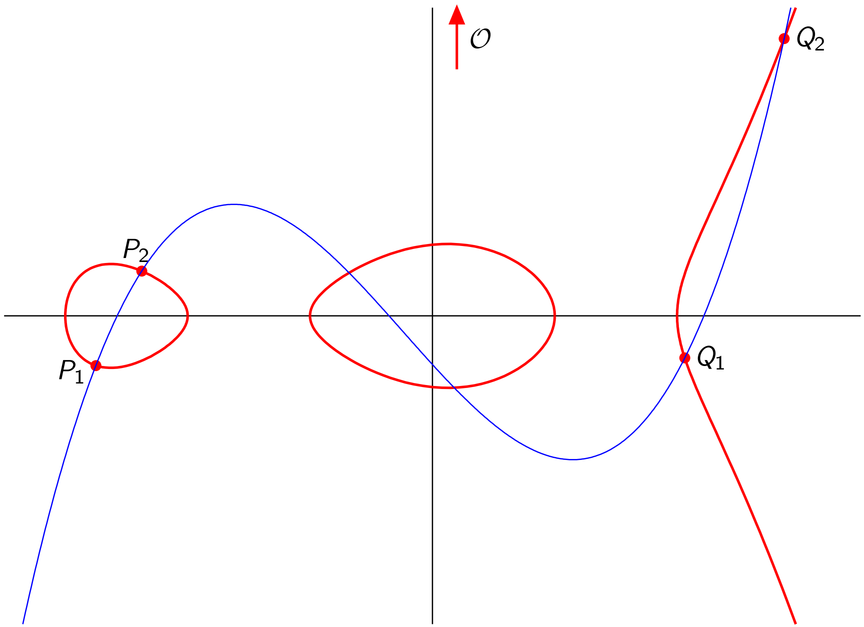


Genus-2 hyperelliptic curves

- $C(K)$ not a group!
- But pairs of points $\{P_1, P_2\}$

$$C/K : y^2 + h(x) \cdot y = f(x)$$

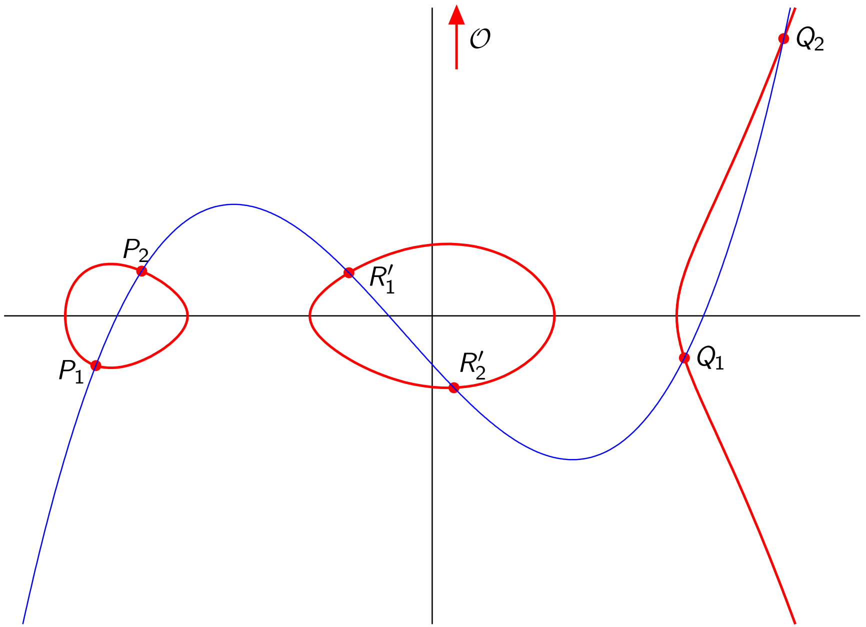
with $\deg h \leq 2$ and $\deg f = 5$



Genus-2 hyperelliptic curves

- $C(K)$ not a group!
- But pairs of points $\{P_1, P_2\}$

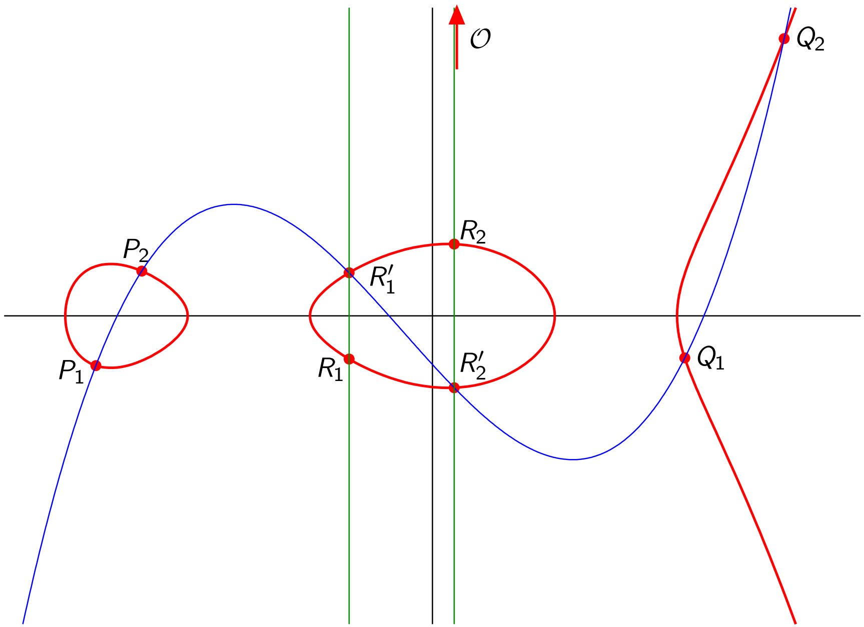
$$C/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with deg $h \leq 2$ and deg $f = 5$

- $C(K)$ not a group!
- But pairs of points $\{P_1, P_2\}$

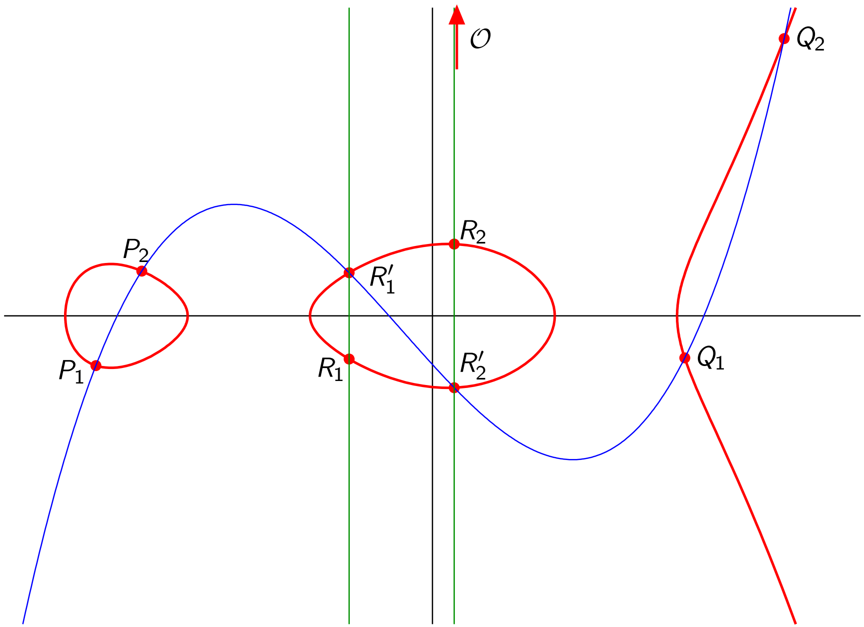


$$\{P_1, P_2\} + \{Q_1, Q_2\} = \{R_1, R_2\}$$

Genus-2 hyperelliptic curves

- $C(K)$ not a group!
- But pairs of points
  $\{P_1, P_2\}$
- More formally
  - use the Jacobian
    $\mathrm{Jac}_C(K)$

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with deg $h \leq 2$ and deg $f = 5$

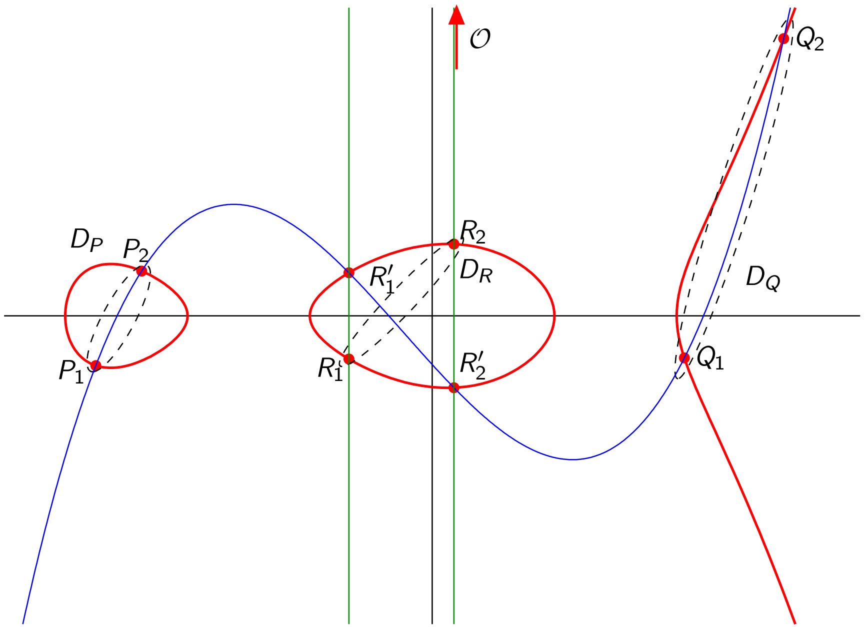


$$\{P_1, P_2\} + \{Q_1, Q_2\} = \{R_1, R_2\}$$

Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

- $C(K)$ not a group!
- But pairs of points $\{P_1, P_2\}$
- More formally
  - use the Jacobian $\mathrm{Jac}_C(K)$

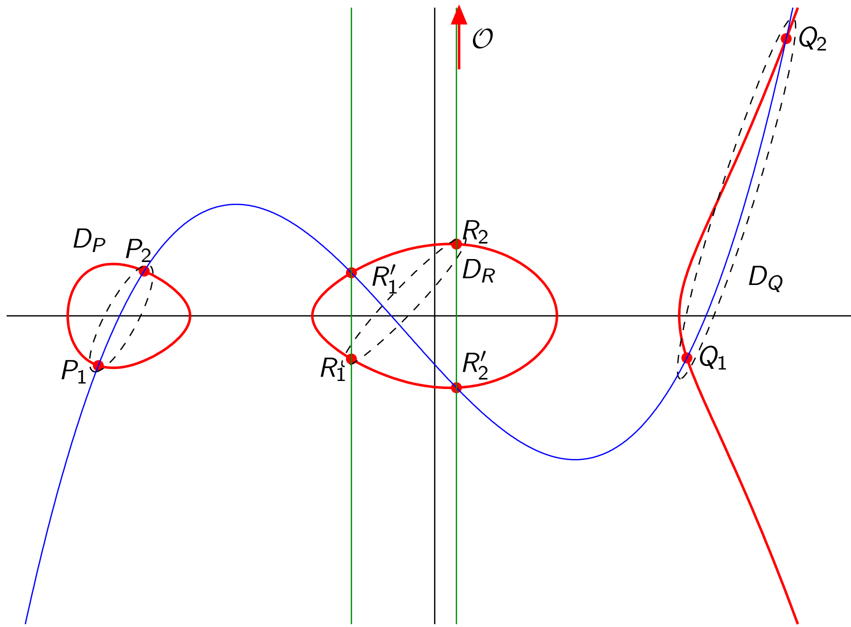


$$D_P + D_Q \qquad = D_R$$

Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with deg $h \leq 2$ and deg $f = 5$

- $C(K)$ not a group!
- But pairs of points
  $\{P_1, P_2\}$
- More formally
  - use the Jacobian
    $\mathrm{Jac}_C(K)$
  - general form of the elements (called divisor)
    $D_P = (P_1) + (P_2) - 2(\mathcal{O})$

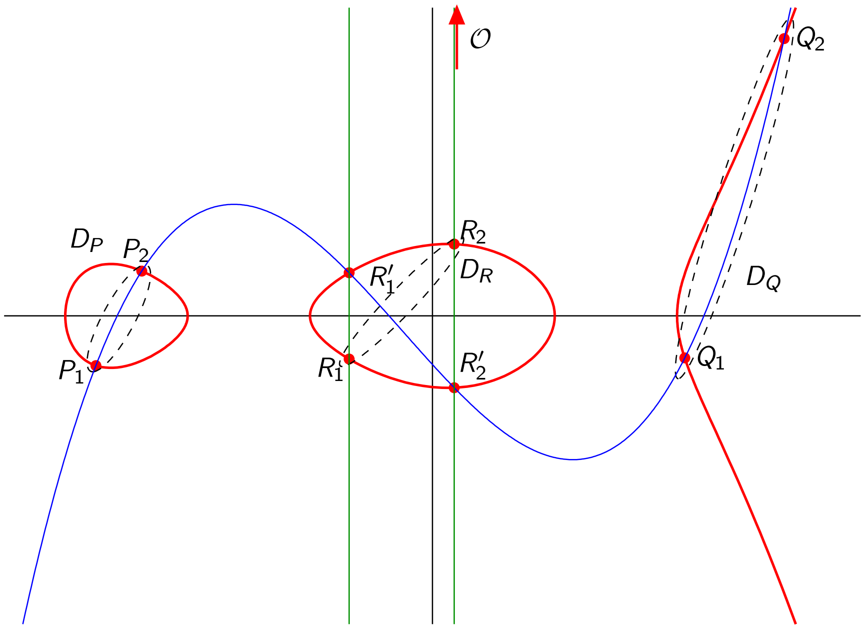


$$D_P + D_Q \qquad = D_R$$

Genus-2 hyperelliptic curves

- $C(K)$ not a group!
- But pairs of points
  $$\{P_1, P_2\}$$
- More formally
  - use the Jacobian
    $$\mathrm{Jac}_C(K)$$
  - general form of the elements (called divisor)
    $$D_P = (P_1)+(P_2)-2(\mathcal{O})$$
  - degenerate form
    $$(P) - (\mathcal{O})$$

$$C/K : y^2 + h(x) \cdot y = f(x)$$
with deg $h \leq 2$ and deg $f = 5$

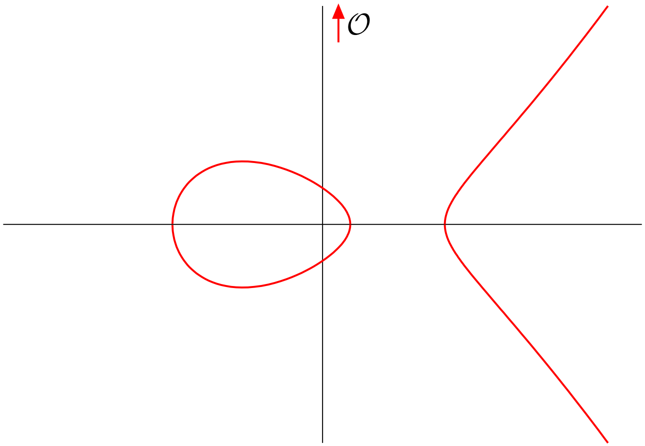


$$D_P + D_Q \qquad = D_R$$

Computing the pairing: Miller's algorithm (elliptic case)

$$e: \quad \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$

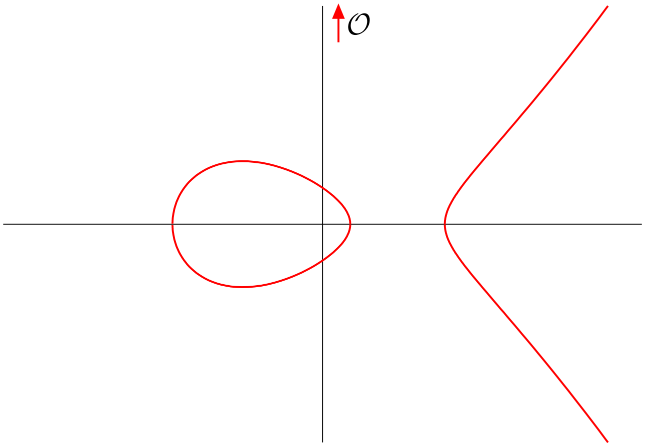
- Reduced Tate pairing



Computing the pairing: Miller's algorithm (elliptic case)

$$\begin{array}{cccccc} e: & E(\mathbb{F}_q)[\ell] & \times & \mathbb{G}_2 & \longrightarrow & \mathbb{G}_T \\ & P & , & & & \end{array}$$

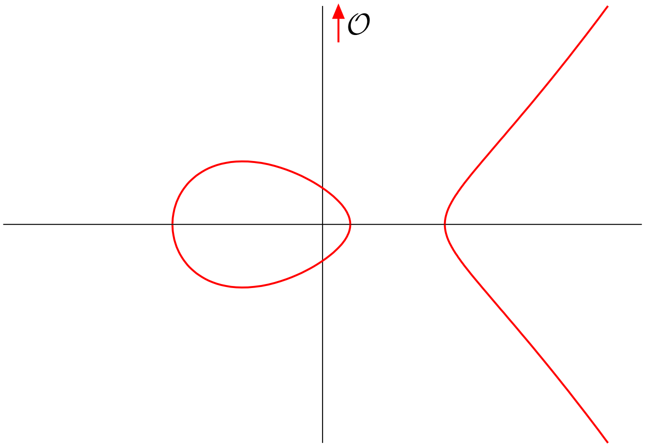
- Reduced Tate pairing



Computing the pairing: Miller's algorithm (elliptic case)

$$\begin{array}{ccccccc} e: & E(\mathbb{F}_q)[\ell] & \times & E(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mathbb{G}_T \\ & P & , & Q & & \end{array}$$

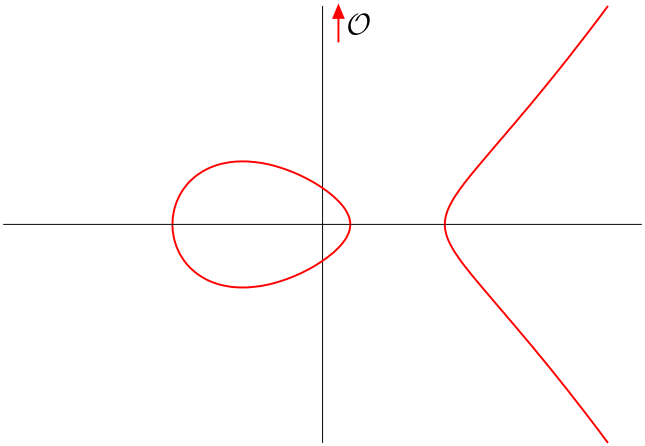
- Reduced Tate pairing
- k: embedding degree (curve parameter)



Computing the pairing: Miller's algorithm (elliptic case)

$$\begin{array}{rcccl} e: E(\mathbb{F}_q)[\ell] & \times & E(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mu_\ell \subset \mathbb{F}_{q^k}^* \\ P & , & Q & \longmapsto & f_{\ell,P}(Q)^{\frac{q^k-1}{\ell}} \end{array}$$

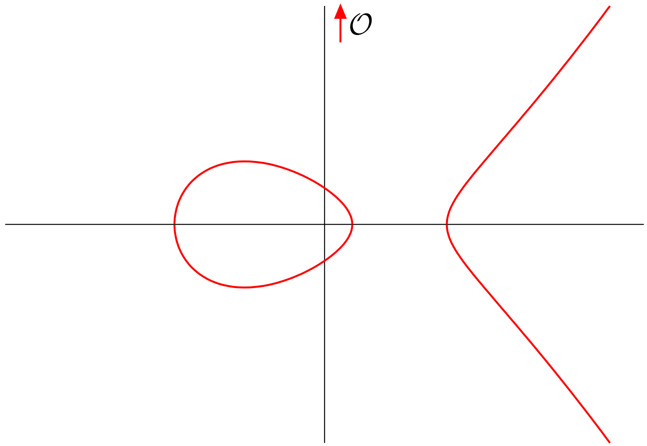
- Reduced Tate pairing
- k: embedding degree (curve parameter)



Computing the pairing: Miller's algorithm (elliptic case)

$$\begin{array}{rcccl} e: & E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mu_\ell \subset \mathbb{F}_{q^k}^* \\ & P \quad , \quad Q & \longmapsto & f_{\ell,P}(Q)^{\frac{q^k-1}{\ell}} \end{array}$$

- Reduced Tate pairing
- k: embedding degree (curve parameter)
- Miller functions: $f_{n,P}$

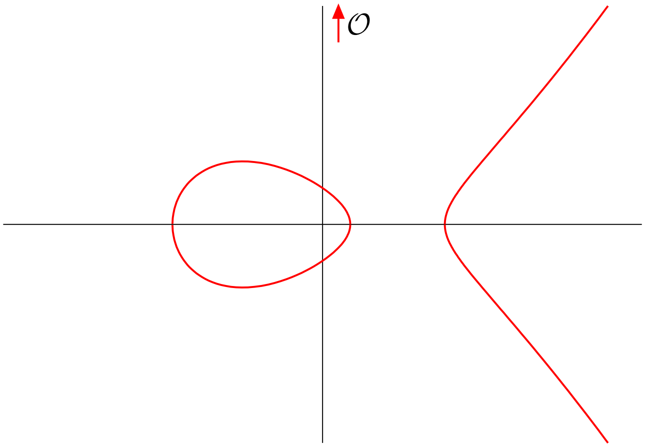


Computing the pairing: Miller's algorithm (elliptic case)

$$\begin{array}{rcccl} e: & E(\mathbb{F}_q)[\ell] & \times & E(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mu_\ell \subset \mathbb{F}_{q^k}^* \\ & P & , & Q & \longmapsto & f_{\ell,P}(Q)^{\frac{q^k-1}{\ell}} \end{array}$$

- Reduced Tate pairing
- k: embedding degree (curve parameter)
- Miller functions: $f_{n,P}$
  - an inductive identity

$$\begin{array}{rcl} f_{1,P} & = & 1 \\ f_{n+n',P} & = & f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{array}$$



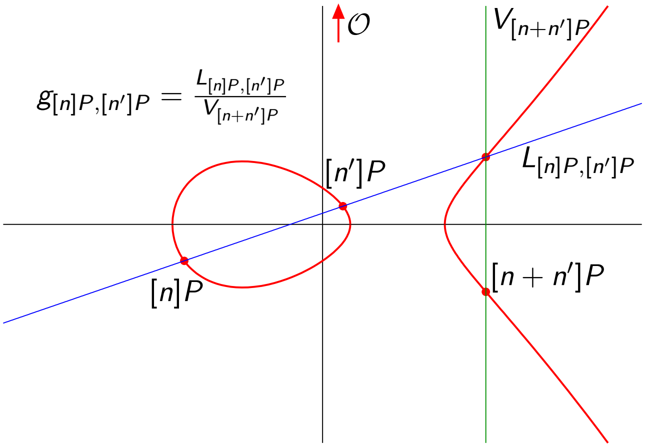
Computing the pairing: Miller's algorithm (elliptic case)

$$\begin{array}{rcccl} e: & E(\mathbb{F}_q)[\ell] & \times & E(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mu_\ell \subset \mathbb{F}_{q^k}^* \\ & P & , & Q & \longmapsto & f_{\ell,P}(Q)^{\frac{q^k-1}{\ell}} \end{array}$$

- Reduced Tate pairing
- k: embedding degree (curve parameter)
- Miller functions: $f_{n,P}$
  - an inductive identity

$$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$

  - $g_{[n]P,[n']P}$ derived from the addition of $[n]P$ and $[n']P$



Computing the pairing: Miller's algorithm (elliptic case)

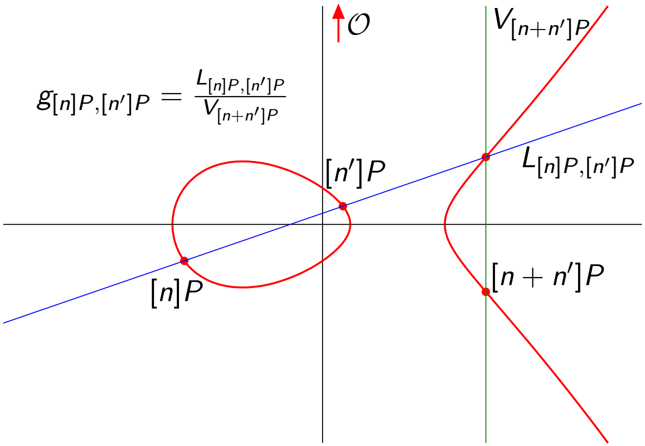
$$\begin{array}{rcccl} e: & E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mu_\ell \subset \mathbb{F}_{q^k}^* \\ & P \quad , \quad Q & \longmapsto & f_{\ell,P}(Q)^{\frac{q^k-1}{\ell}} \end{array}$$

- Reduced Tate pairing
- k: embedding degree (curve parameter)
- Miller functions: $f_{n,P}$
  - an inductive identity

$$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$

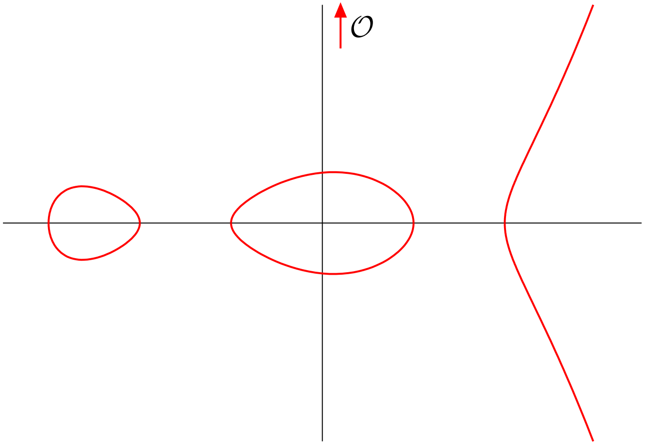
  - $g_{[n]P,[n']P}$ derived from the addition of $[n]P$ and $[n']P$
  - compute $f_{\ell,P}$ thanks to an addition chain
  - in practice: double-and-add

$\log_2 \ell$ iterations



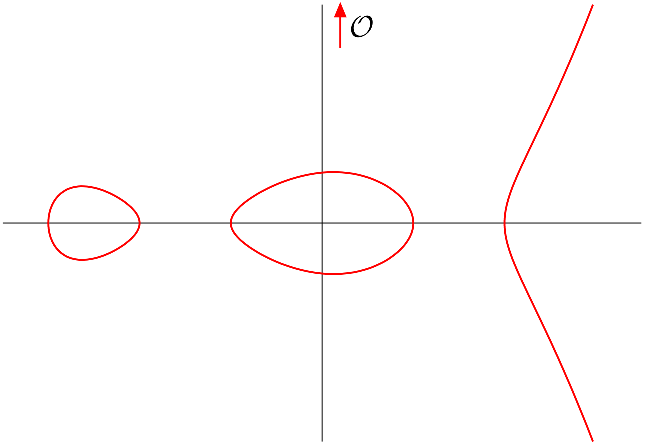
Miller's algorithm (hyperelliptic case)

$$e: \quad \mathbb{G}_1 \times \mathbb{G}_2 \longrightarrow \mathbb{G}_T$$



Miller's algorithm (hyperelliptic case)

$$e: \quad \mathrm{Jac}_C(\mathbb{F}_q)[\ell] \;\times\; \mathrm{Jac}_C(\mathbb{F}_{q^k})[\ell] \longrightarrow \mu_\ell \subset \mathbb{F}_{q^k}^*$$

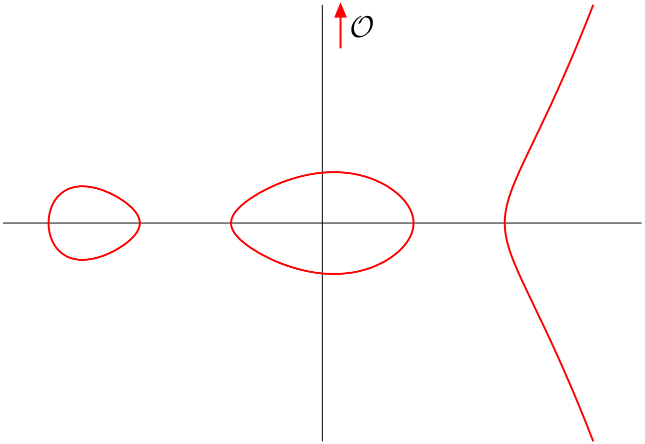


Miller's algorithm (hyperelliptic case)

$$\begin{array}{rcccl} e: & \mathrm{Jac}_C(\mathbb{F}_q)[\ell] & \times & \mathrm{Jac}_C(\mathbb{F}_{q^k})[\ell] & \longrightarrow & \mu_\ell \subset \mathbb{F}_{q^k}^* \\ & D_1 & , & D_2 & \longmapsto & f_{\ell,D_1}(D_2)^{\frac{q^k-1}{\ell}} \end{array}$$

- Hyperelliptic Miller functions: $f_{n,D}$
  - same inductive identity

$$\begin{aligned} f_{1,D} &= 1 \\ f_{n+n',D} &= f_{n,D} \cdot f_{n',D} \cdot g_{[n]D,[n']D} \end{aligned}$$



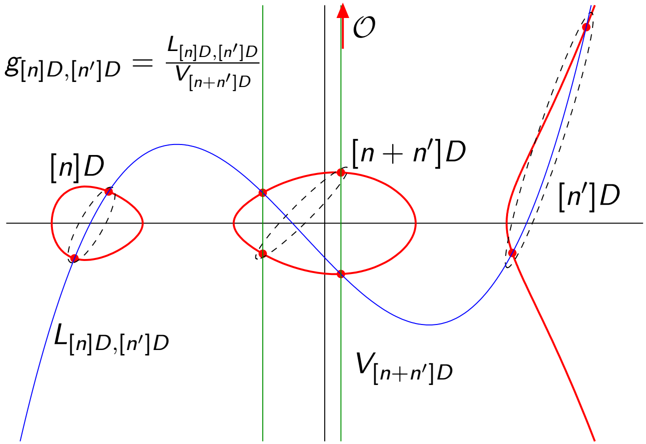
Miller's algorithm (hyperelliptic case)

$$\begin{array}{rcccl} e: & \mathrm{Jac}_C(\mathbb{F}_q)[\ell] & \times & \mathrm{Jac}_C(\mathbb{F}_{q^k})[\ell] & \longrightarrow \mu_\ell \subset \mathbb{F}_{q^k}^* \\ & D_1 & , & D_2 & \longmapsto f_{\ell,D_1}(D_2)^{\frac{q^k-1}{\ell}} \end{array}$$

- Hyperelliptic Miller functions: $f_{n,D}$
  - same inductive identity

$$\begin{aligned} f_{1,D} &= 1 \\ f_{n+n',D} &= f_{n,D} \cdot f_{n',D} \cdot g_{[n]D,[n']D} \end{aligned}$$

  - $g_{[n]D,[n']D}$ derived from the addition of $[n]D$ and $[n']D$
  - use Cantor's addition algorithm



Miller's algorithm (hyperelliptic case)

$$
\begin{array}{rcccl}
e: & \mathrm{Jac}_C(\mathbb{F}_q)[\ell] & \times & \mathrm{Jac}_C(\mathbb{F}_{q^k})[\ell] & \longrightarrow \mu_\ell \subset \mathbb{F}_{q^k}^* \\
 & D_1 & , & D_2 & \longmapsto f_{\ell,D_1}(D_2)^{\frac{q^k-1}{\ell}}
\end{array}
$$

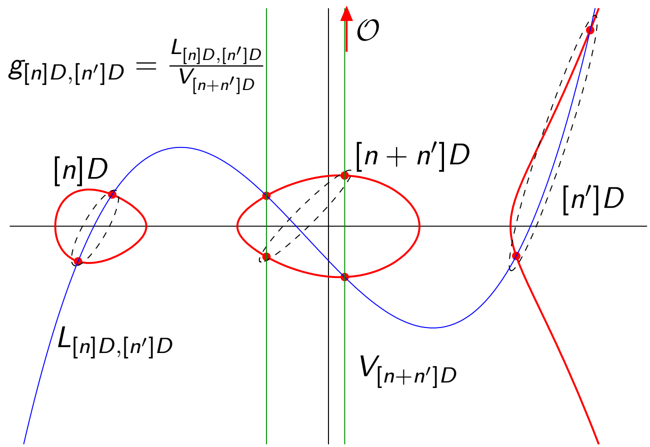
- Hyperelliptic Miller functions: $f_{n,D}$
  - same inductive identity

$$
\begin{aligned}
f_{1,D} &= 1 \\
f_{n+n',D} &= f_{n,D} \cdot f_{n',D} \cdot g_{[n]D,[n']D}
\end{aligned}
$$

  - $g_{[n]D,[n']D}$ derived from the addition of $[n]D$ and $[n']D$
  - use Cantor's addition algorithm
  - double-and-add algorithm

    $\log_2 \ell$ iterations

  - iterations are more complex



Genus-2 binary supersingular curve: our choice

$$C_d/\mathbb{F}_{2^m} : y^2 + y = x^5 + x^3 + d \text{ with } d \in \mathbb{F}_2$$

- A distortion map exists: symmetric pairing
- $\#\,\mathrm{Jac}_{C_d}(\mathbb{F}_{2^m}) = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$
- Embedding degree of the curve: $k = 12$
- For 128 bits of security: $\mathbb{F}_{2^m} = \mathbb{F}_{2^{367}}$ and $d = 0$

Genus-2 binary supersingular curve: our choice

$$C_d/\mathbb{F}_{2^m} : y^2 + y = x^5 + x^3 + d \text{ with } d \in \mathbb{F}_2$$

- A distortion map exists: symmetric pairing
- $\# \operatorname{Jac}_{C_d}(\mathbb{F}_{2^m}) = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$
- Embedding degree of the curve: $k = 12$
- For 128 bits of security: $\mathbb{F}_{2^m} = \mathbb{F}_{2^{367}}$ and $d = 0$
- Key property of the curve:

$$[2]\left((P) - (\mathcal{O})\right) \;=\; (P) + (P) - 2(\mathcal{O})$$

Genus-2 binary supersingular curve: our choice

$$C_d/\mathbb{F}_{2^m} : y^2 + y = x^5 + x^3 + d \text{ with } d \in \mathbb{F}_2$$

- A distortion map exists: symmetric pairing
- $\# \mathrm{Jac}_{C_d}(\mathbb{F}_{2^m}) = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$
- Embedding degree of the curve: $k = 12$
- For 128 bits of security: $\mathbb{F}_{2^m} = \mathbb{F}_{2^{367}}$ and $d = 0$
- Key property of the curve:

$$\begin{aligned} [2]\left((P) - (\mathcal{O})\right) &= (P) + (P) - 2(\mathcal{O}) \\ [4]\left((P) - (\mathcal{O})\right) &= (P_4) + (P_4') - 2(\mathcal{O}) \end{aligned}$$

Genus-2 binary supersingular curve: our choice

$$C_d/\mathbb{F}_{2^m} : y^2 + y = x^5 + x^3 + d \text{ with } d \in \mathbb{F}_2$$

- A distortion map exists: symmetric pairing
- $\# \operatorname{Jac}_{C_d}(\mathbb{F}_{2^m}) = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$
- Embedding degree of the curve: $k = 12$
- For 128 bits of security: $\mathbb{F}_{2^m} = \mathbb{F}_{2^{367}}$ and $d = 0$
- Key property of the curve:

$$\begin{aligned}
[2]\left((P) - (\mathcal{O})\right) &= (P) + (P) - 2(\mathcal{O}) \\
[4]\left((P) - (\mathcal{O})\right) &= (P_4) + (P_4') - 2(\mathcal{O}) \\
[8]\left((P) - (\mathcal{O})\right) &= (P_8) \quad - (\mathcal{O})
\end{aligned}$$

Genus-2 binary supersingular curve: our choice

$$C_d/\mathbb{F}_{2^m} : y^2 + y = x^5 + x^3 + d \text{ with } d \in \mathbb{F}_2$$

- A distortion map exists: symmetric pairing
- $\# \mathrm{Jac}_{C_d}(\mathbb{F}_{2^m}) = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$
- Embedding degree of the curve: $k = 12$
- For 128 bits of security: $\mathbb{F}_{2^m} = \mathbb{F}_{2^{367}}$ and $d = 0$
- Key property of the curve:

$$\begin{aligned}
[2]\,((P) - (\mathcal{O})) &= (P) + (P) - 2(\mathcal{O}) \\
[4]\,((P) - (\mathcal{O})) &= (P_4) + (P_4') - 2(\mathcal{O}) \\
[8]\,((P) - (\mathcal{O})) &= ([8]P) - (\mathcal{O})
\end{aligned}$$

  - octupling acts on the curve

Genus-2 binary supersingular curve: our choice

$$C_d/\mathbb{F}_{2^m} : y^2 + y = x^5 + x^3 + d \text{ with } d \in \mathbb{F}_2$$

- A distortion map exists: symmetric pairing
- $\# \mathrm{Jac}_{C_d}(\mathbb{F}_{2^m}) = 2^{2m} \pm 2^{(3m+1)/2} + 2^m \pm 2^{(m+1)/2} + 1$
- Embedding degree of the curve: $k = 12$
- For 128 bits of security: $\mathbb{F}_{2^m} = \mathbb{F}_{2^{367}}$ and $d = 0$
- Key property of the curve:

$$\begin{aligned}
[2]\,((P) - (\mathcal{O})) &= (P) + (P) - 2(\mathcal{O}) \\
[4]\,((P) - (\mathcal{O})) &= (P_4) + (P'_4) - 2(\mathcal{O}) \\
[8]\,((P) - (\mathcal{O})) &= ([8]P) - (\mathcal{O})
\end{aligned}$$

  - octupling acts on the curve
  - $f_{8,D}$ has a much simpler expression than $f_{2,D}$

Constructing the Optimal Eta pairing

Algorithm	Tate double & add			
#iterations	$2m$			

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings

Constructing the Optimal Eta pairing

Algorithm	Tate double & add	Tate octuple & add		
#iterations	$2m$	$\frac{2m}{3}$		

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings
- Use of octupling: simpler iteration also!

Constructing the Optimal Eta pairing

Algorithm	Tate double & add	Tate octuple & add	Barreto *et al.* η_T pairing	
#iterations	$2m$	$\frac{2m}{3}$	$\frac{m}{2}$	

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings
- Use of octupling: simpler iteration also!
- η_T pairing: Miller function is $f_{\pm 2^{(3m+1)/2}-1, D_1}$

Constructing the Optimal Eta pairing

Algorithm	Tate double & add	Tate octuple & add	Barreto *et al.* η_T pairing	Optimal Ate pairing
#iterations	$2m$	$\frac{2m}{3}$	$\frac{m}{2}$	$\frac{m}{6}$

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings
- Use of octupling: simpler iteration also!
- η_T pairing: Miller function is $f_{\pm 2^{(3m+1)/2}-1, D_1}$
- Optimal Ate pairing
  - distortion map ψ is much more complex
  - iterations would be roughly twice as expensive
  - optimal Ate pairing not considered here

Constructing the Optimal Eta pairing

Algorithm	Tate double & add	Tate octuple & add	Barreto *et al.* η_T pairing	This paper Optimal Eta pairing
#iterations	$2m$	$\frac{2m}{3}$	$\frac{m}{2}$	$\frac{m}{6}$

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings
- Use of octupling: simpler iteration also!
- η_T pairing: Miller function is $f_{\pm 2^{(3m+1)/2}-1, D_1}$
- Optimal Ate pairing
  - distortion map ψ is much more complex
  - iterations would be roughly twice as expensive
  - optimal Ate pairing not considered here
- Optimal Eta pairing

Constructing the Optimal Eta pairing

Algorithm	Tate double & add	Tate octuple & add	Barreto *et al.* η_T pairing	This paper Optimal Eta pairing
#iterations	$2m$	$\frac{2m}{3}$	$\frac{m}{2}$	$\frac{m}{6}$

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings
- Use of octupling: simpler iteration also!
- η_T pairing: Miller function is $f_{\pm 2^{(3m+1)/2}-1, D_1}$
- Optimal Ate pairing
  - distortion map ψ is much more complex
  - iterations would be roughly twice as expensive
  - optimal Ate pairing not considered here
- Optimal Eta pairing
  - cannot use 2^m-th power Verschiebung: does not act on the curve

Constructing the Optimal Eta pairing

Algorithm	Tate double & add	Tate octuple & add	Barreto *et al.* η_T pairing	This paper Optimal Eta pairing
#iterations	$2m$	$\frac{2m}{3}$	$\frac{m}{2}$	$\frac{m}{3}$

- Vanilla Tate pairing: $\log_2 \ell \approx \log_2 \# \operatorname{Jac}_C(\mathbb{F}_{2^m}) \approx 2m$ doublings
- Use of octupling: simpler iteration also!
- η_T pairing: Miller function is $f_{\pm 2^{(3m+1)/2} - 1, D_1}$
- Optimal Ate pairing
  - distortion map ψ is much more complex
  - iterations would be roughly twice as expensive
  - optimal Ate pairing not considered here
- Optimal Eta pairing
  - cannot use 2^m-th power Verschiebung: does not act on the curve
  - but can use 2^{3m}-th power Verschiebung
  - 33% improvement compared to Barreto *et al.*'s work

Considering degenerate divisors

- Some protocols allow to choose the form of one or two input divisors
- Consider degenerate divisors of the form

$$(P) - (\mathcal{O})$$

  - only 2 coordinates in $\mathbb{F}_{2^m}$ to represent such a divisor (instead of 4 coordinates for a general one)
  - since octupling acts on the curve:

$$[8]((P) - (\mathcal{O})) = ([8]P) - (\mathcal{O})$$

  - we can work with a point!

Considering degenerate divisors

- Some protocols allow to choose the form of one or two input divisors
- Consider degenerate divisors of the form

$$(P) - (\mathcal{O})$$

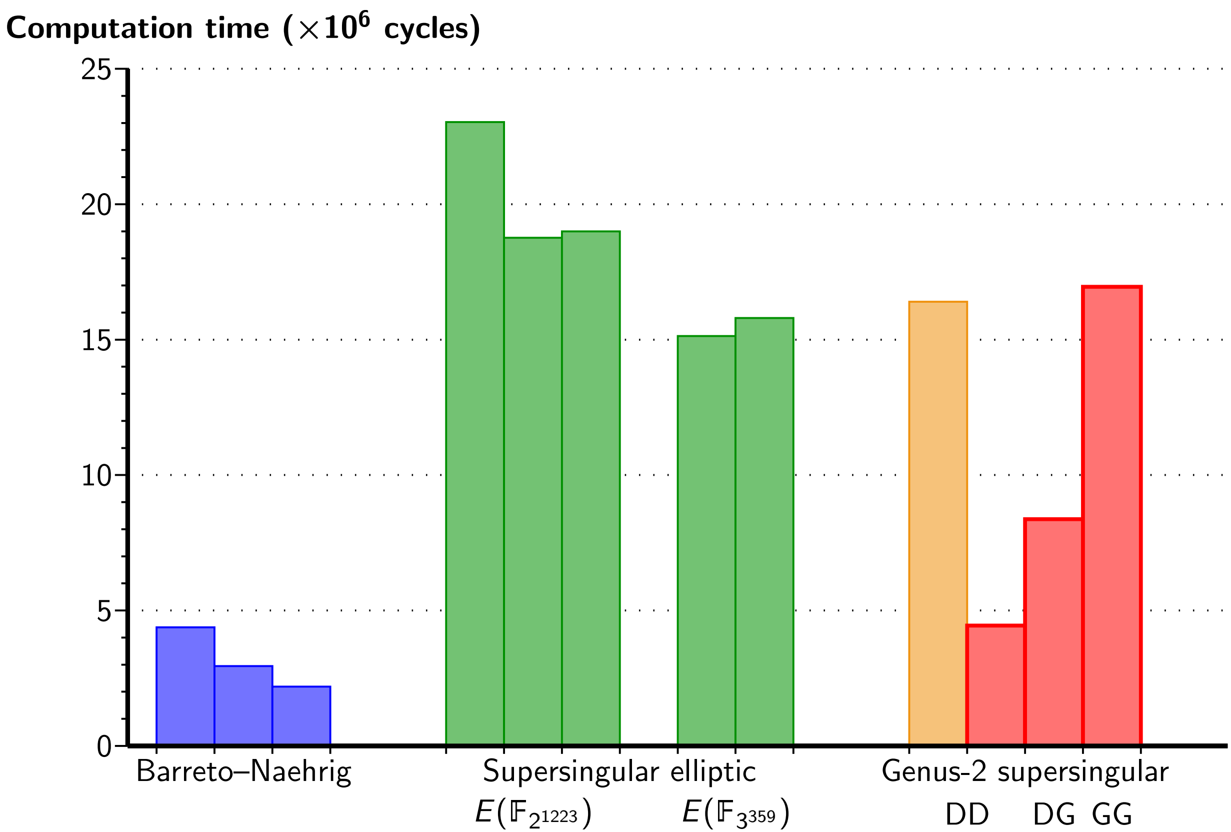
  - only 2 coordinates in $\mathbb{F}_{2^m}$ to represent such a divisor (instead of 4 coordinates for a general one)
  - since octupling acts on the curve:

$$[8]((P) - (\mathcal{O})) = ([8]P) - (\mathcal{O})$$

  - we can work with a point!
- We may compute the pairing of
  - two general divisors (GG)
  - one degenerate and one general divisor (DG)
    - ⋆ halves the amount of computation
    - ⋆ lot of protocols allow this
  - two degenerate divisors (DD)
    - ⋆ halves again the amount of computation
    - ⋆ some protocols still compatible

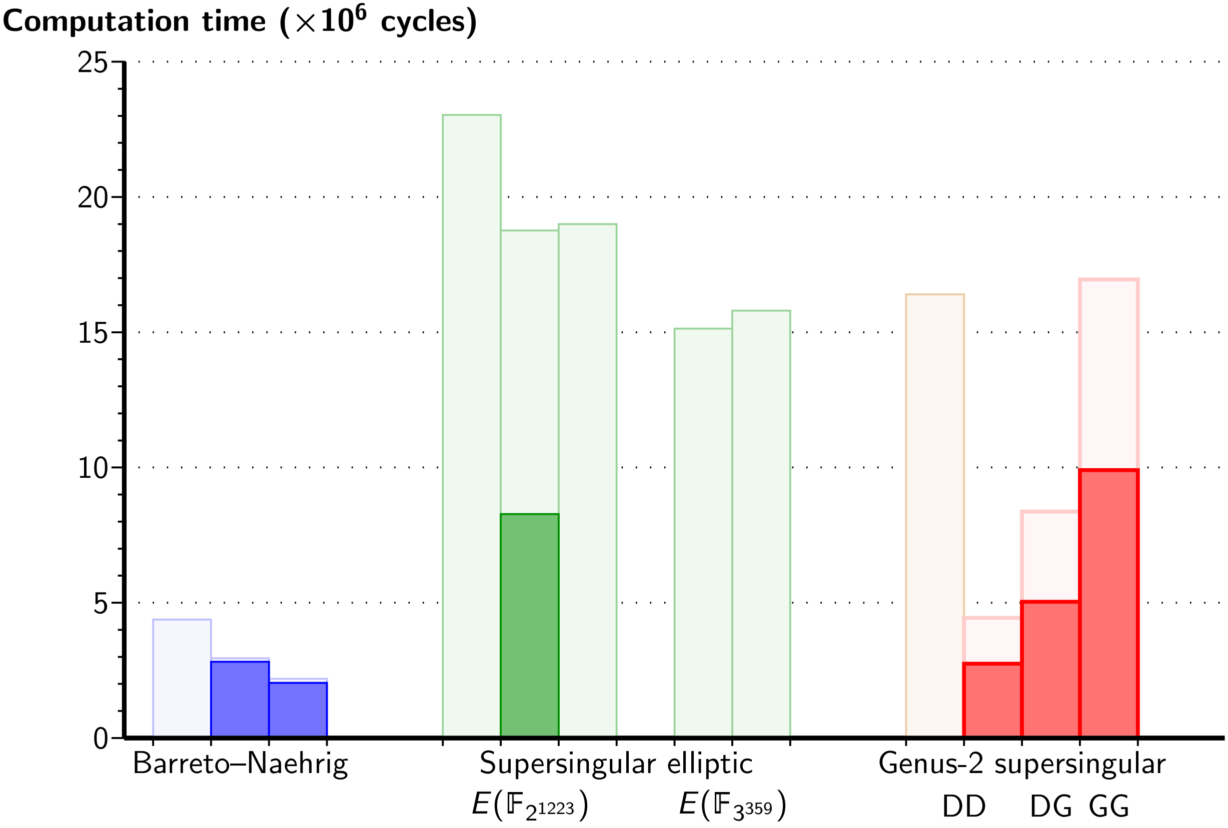
Software implementation

- Implementations for Intel Core 2



Software implementation

- Implementations for Intel Core 2 and Nehalem architecture
- Use of the native binary field multiplier on Nehalem



Hardware implementation

- Optimal Eta pairing on general divisors
- Implemented on a finite field coprocessor $\mathbb{F}_{2^{367}}$
  - addition
  - multiplication
  - Frobenius endomorphism
- Post place-and-route estimations on a Virtex 6-LX 130T results

Implementation	**Curve**	**Area (device usage)**	**Time (ms)**	**Area×time**
Cheung *et al.*	$E(\mathbb{F}_{p_{254}})$	35 %	0.57	4.03
Ghosh *et al.*	$E(\mathbb{F}_{2^{1223}})$	76 %	0.19	2.88
Estibals	$E(\mathbb{F}_{3^{5\cdot 97}})$	8 %	1.73	2.68
This work	$C_0(\mathbb{F}_{2^{367}})$ (GG)	7 %	3.09	4.30

Conclusion

- A novel pairing algorithm shortening Miller's loop
- Competitive timings compared to genus-1 pairings
- Comparable timings against non-symmetric pairings

Conclusion

- A novel pairing algorithm shortening Miller's loop
- Competitive timings compared to genus-1 pairings
- Comparable timings against non-symmetric pairings
- Most efficient symmetric pairing implementation
  - for both software and hardware
  - when at least one divisor is degenerate (DG and DD case)
- First hardware implementation of a genus-2 pairing reaching 128 bits of security

Conclusion

- A novel pairing algorithm shortening Miller's loop
- Competitive timings compared to genus-1 pairings
- Comparable timings against non-symmetric pairings
- Most efficient symmetric pairing implementation
  - for both software and hardware
  - when at least one divisor is degenerate (DG and DD case)
- First hardware implementation of a genus-2 pairing reaching 128 bits of security
- Perspectives
  - Implement optimal Ate pairing on this curve (work in progress)
  - Use theta functions for faster curve arithmetic

Thank you for your attention!

Questions?