A Formal Library for Elliptic Curves in the Coq Proof Assistant

Evmorfia-Iro Bartzia, Pierre-Yves Strub

A Formal Library for Elliptic Curves in the Coq Proof Assistant

Evmorfia-Iro Bartzia[1] and Pierre-Yves Strub[2]

[1] INRIA Paris-Rocquencourt, France
iro.bartzia@inria.fr
[2] IMDEA Software Institute, Spain
pierre-yves@strub.nu

Abstract. A preliminary step towards the verification of elliptic curve cryptographic algorithms is the development of formal libraries with the corresponding mathematical theory. In this paper we present a formalization of elliptic curves theory, in the SSReflect extension of the Coq proof assistant. Our central contribution is a library containing many of the objects and core properties related to elliptic curve theory. We demonstrate the applicability of our library by formally proving a non-trivial property of elliptic curves: the existence of an isomorphism between a curve and its Picard group of divisors.

1 Introduction

The design of cryptographic algorithms is a complicated task. Besides functional correctness, cryptographic algorithms need to achieve contradictory goals such as efficiency and side channel resistance. Faulty implementations of algorithms may endanger security [4]. This is why formal assurance about their correctness is essential. Our motivation is to develop libraries that allow the formal verification of asymmetric cryptographic algorithms. As of today, the work on formal verification of security protocols has been assuming that the cryptographic libraries correctly implement all algorithms [2]. The first step towards the formal verification of cryptographic algorithms is the development of libraries that formally express the corresponding mathematical theory. In this paper we present a formal library for elementary elliptic curve theory that will enable formal analysis of elliptic-curve algorithms.

Elliptic curves have been used since the 19th century to approach a wide range of problems such as the fast factorization of integers and the search for congruent numbers. In the 20th century, researchers have regained interest in elliptic curves because of their applications in cryptography, first suggested in 1985 independently by Neal Koblitz [14] and Victor Miller [15]. Their use in cryptography relies principally on the existence of a group law that is a good candidate for public key cryptography, as its Discrete Logarithm Problem is hard relatively to the size of the parameters used. Elliptic curves also allow the definition of digital signatures and of new cryptographic primitives, such

as identity-based encryption [17], based on bilinear (Weil and Tate) pairings. The mathematics of elliptic curves used in cryptography start from defining the group law and continue to theory from algebraic geometry [9].

Because our formalization involves algebraic structures such as rings and groups, polynomials, rational functions and matrices, we use the SSReflect extension [11] of the Coq proof-assistant [19] and its mathematical components library [1]. The Coq development can be found on the second author website (`http://pierre-yves.strub.nu/`).

Contributions. This paper presents an attempt to formalize non-trivial objects of algebraic geometry such as elliptic curves, rational functions and divisors. Our library is designed in such a way that will enable formal proofs of functional correctness of elliptic-curve algorithms. We validate the applicability of our theory by formally proving the Picard theorem, i.e. that an elliptic curve is structurally equivalent with its Picard group of divisors. Our formalization follows an elementary proof from Guillot [12] and Charlap [5].

Paper Outline. In sections 2 to 4, we present a formal proof of the following proposition, referred later as the *Picard theorem*:

> *The set of points of an elliptic curve together with its operation is isomorphic to its Picard group of divisors.*

We first define the two structures - namely the elliptic curve (Section 2) and the Picard group (Section 3) - and then prove that there exists a group isomorphism between them. In contrast to the definition of an elliptic curve, which goes smoothly, the definition of the Picard group involves several steps and forms the main matter of this paper. By construction, the Picard group of divisors is equipped with a group structure. In Section 4, we prove that the two structures are isomorphic. By transport of structure, the set of points of an elliptic curve together with its operation forms a group. In Section 5 and 6, we discuss related and future work.

2 Formalizing Elliptic Curves

An elliptic curve is a special case of a projective algebraic curve that can be defined as follows:

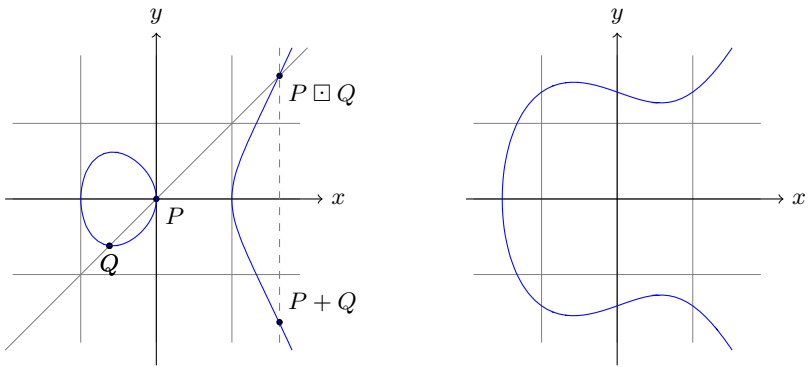
Definition 1. *Let $\mathbb{K}$ be a field. Using an appropriate choice of coordinates, an* elliptic curve *$\mathcal{E}$ is a plane cubic algebraic curve $\mathcal{E}(x, y)$ defined by an equation of the form:*

$$\mathcal{E}\colon y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

where the a_i's are in $\mathbb{K}$ and the curve has no singular point (i.e. no cusps or self-intersections). The set of points, written $\mathcal{E}(\mathbb{K})$, is formed by the solutions (x, y) of $\mathcal{E}$ augmented by a distinguished point $\mathcal{O}$ (called point at infinity*):*

$$\mathcal{E}(\mathbb{K}) = \{(x, y) \in \mathbb{K} \mid \mathcal{E}(x, y)\} \cup \{\mathcal{O}\}$$

Figure 1 provides graphical representations of such curves in the real plane.



(a) $y^2 = x^3 + x$ (b) $y^2 = x^3 - x + 2$

Fig. 1: Catalog of Elliptic Curves Graphs

When the characteristic of $\mathbb{K}$ is different from 2 and 3, the equation $\mathcal{E}(x, y)$ can be simplified into its *Weierstrass* form:

$$y^2 = x^3 + ax + b.$$

Moreover, such a curve does not present any singularity if $\Delta(a, b) = 4a^3 + 27b^2$ — the curve's discriminant — is not equal to 0. Our work lies in this setting.

The parametric type `ec` represents the points on a specific curve. It is parameterized by a `K : ecuFieldType` — the type of fields with characteristic not in $\{2, 3\}$ — and a `E : ecuType` — a record that packs the curve parameters a and b along with a proof that $\Delta(a, b) \neq 0$. An inhabitant of the type `ec` is a point of the projective plane (represented by the type `point`), along with a proof that the point is on the curve.

```
Record ecuType := { A : K; B : K; _ : 4 * A^3 + 27 * B^2 != 0 }.
Inductive point := EC_Inf | EC_In of K & K.
Notation "(x, y)" := (EC_In x y).
Definition oncurve (p : point) :=
  if p is (x, y) then y^2 == x^3 + A * x + B else true.
Inductive ec : Type := EC p of oncurve p.
```

The points of an elliptic curve can be equipped with a structure of an abelian group. We give here a geometrical construction of the law. Let P and Q be points on the curve $\mathcal{E}$ and l be the line that goes through P and Q (or that is tangent

to the curve at P if $P = Q$). By the Bezout theorem, counting multiplicities, l intersects $\mathcal{E}$ at a third point, denoted by $P \boxdot Q$. The sum $P + Q$ is the opposite of $P \boxdot Q$, obtained by taking the symmetric of $P \boxdot Q$ with respect to the x axis. Figure 1 highlights this construction. To sum up:

1. $\mathcal{O}$ is defined to be the neutral element: $\forall P.\, P + \mathcal{O} = \mathcal{O} + P = P$,
2. the opposite of a point (x_P, y_P) (resp. $\mathcal{O}$) is $(x_P, -y_P)$ (resp. $\mathcal{O}$), and
3. if three points are collinear, their sum is equal to $\mathcal{O}$.

This geometrical definition can be translated into an algebraic setting, obtaining polynomial formulas for the definition of the law. Having such polynomial formulas leads to the following definitions:

```
Definition neg (p : point) :=
  if p is (x, y) then (x, -y) else EC_Inf.

Definition add (p1 p2 : point) :=
  let p1 := if oncurve p1 then p1 else EC_Inf in
  let p2 := if oncurve p2 then p2 else EC_Inf in
    match p1, p2 with
    | EC_Inf, _ => p2 | _, EC_Inf => p1
    | (x1, y1), (x2, y2) =>
      if x1 == x2 then ... else
        let s := (y2 - y1) / (x2 - x1) in
        let xs := s^2 - x1 - x2 in
          (xs, - s * (xs - x1) - y1)
    end.
```

Note that these definitions do not directly work with points on the curve, but instead on points of the projective plane (points that do not lie on the curve are projected to $\mathcal{O}$). We then prove that these operations are internal to the curve and lift them to $\mathcal{E}$:

```
Lemma add0 (p q : point): oncurve (add p q).
Definition addec (p1 p2 : ec) : ec := EC p1 p2 (add0 p1 p2).
```

We link back this algebraic definition to its geometrical interpretation. First, we define a function `line`: given two points P, Q on the curve, it returns the equation $ux + vy + c = 0$ of the line (PQ) intersecting the curve at P and Q (resp. the equation of the tangent to the curve at P if $P = Q$). We then show that, if (PQ) is not parallel to the y axis (i.e. is not intersecting the curve at $\mathcal{O}$), then (PQ) is intersecting $\mathcal{E}$ exactly at P, Q and $-(P + Q) = P \boxdot Q$ as defined algebraically. This proof mainly relies on Vieta's formulas that relate the coefficients of a polynomial to sums and products of its roots. Although only a specific instance of Vieta's formulas is needed, we formalized the general ones:

Lemma 1 (Vieta's formulas). *For any polynomial $p = \sum_{i \leq n} a_i X^i$ with roots $x_1, \ldots, x_n$, over an algebraically closed field, we have:*

$$\forall k.\, \sigma_k(x_1, \ldots, x_n) = (-1)^k \cdot \frac{a_{n-k}}{a_n}$$

where σ_k is the k^{th}-elementary symmetrical polynomial.

3 The Picard Group of Divisors

From now on, let $\mathcal{E}$ be a smooth elliptic curve with equation $y^2 = x^3 + ax + b$ over the field $\mathbb{K}$. We assume that $\mathbb{K}$ is not of characteristic 2, nor 3. Related to this curve, we assume two Coq parameters `K : ecuFieldType` and `E : ecuType K`. We now move to the construction of the *Picard group* $\mathrm{Pic}(\mathcal{E})$. This construction is split into several steps:

1. We start by constructing two objects: the field of rational functions $\mathbb{K}(\mathcal{E})$ over $\mathcal{E}$ and the group of $\mathcal{E}$-divisors $\mathrm{Div}(\mathcal{E})$, i.e. the set of formal sums over the points of $\mathcal{E}$. From $\mathrm{Div}(\mathcal{E})$ we construct $\mathrm{Div}^0(\mathcal{E})$ which is the subgroup of zero-degree divisors.
2. We attach to each rational function $f \in \mathbb{K}(\mathcal{E})$ a divisor $\mathrm{Div}(f)$ (called *principal divisor*) that characterizes f up to a scalar multiplication. This allows us to define the subgroup $\mathrm{Prin}(\mathcal{E})$ of $\mathrm{Div}(\mathcal{E})$, namely the *group of principal divisors*. The quotient group $\mathrm{Div}^0(\mathcal{E})/\mathrm{Prin}(\mathcal{E})$ forms the *Picard group*.

3.1 The field of rational functions $\mathbb{K}(\mathcal{E})$

We denote the ring of bivariate polynomials over $\mathbb{K}$ by $\mathbb{K}[x, y]$.

Definition 2. *The ring $\mathbb{K}[\mathcal{E}]$ of* polynomials over the curve *is defined as the quotient ring of $\mathbb{K}[x, y]$ by the prime ideal $\langle y^2 - (x^3 + ax + b)\rangle$. The field $\mathbb{K}(\mathcal{E})$ is defined as the field of fractions of the integral domain $\mathbb{K}[\mathcal{E}]$.*

In other words, $\mathbb{K}[\mathcal{E}]$ is defined as the quotient of $\mathbb{K}[x, y]$ by the following equivalence relation $\sim$:

$$p \sim q \text{ if and only if } \exists k \in \mathbb{K}[x, y] \text{ such that } p - q = k(y^2 - x^3 - ax - b).$$

Since the polynomials y^2 and $x^3 + ax + b$ are identified in $\mathbb{K}[\mathcal{E}]$, we can associate, to any equivalence class of $\mathbb{K}[\mathcal{E}]$, a canonical representative of the form $p_1 y + p_2$ ($p_1, p_2 \in \mathbb{K}[x]$), obtained by iteratively substituting y^2 by $x^3 + ax + b$ in any element of the equivalence class. As such, instead of going through the path of formalizing ideals and ring quotients, we give a direct representation of $\mathbb{K}[\mathcal{E}]$ solely based on `{poly K}`, the type of univariate polynomials over `K`:

```
Inductive ecring := ECRing of {poly K} * {poly K}.
Notation "[ecp p1 *Y + p2]" := (ECRing p1 p2).
Coercion ecring_val (p : ecring) := let: ECRing p := p in p.
```

The type `ecring` is simply a copy of `{poly K} * {poly K}`, an element `([ecp p1 *Y + p2] : ecring)` representing the class of the polynomial $p_1 y + p_2 \in \mathbb{K}[\mathcal{E}]$. We explicitly define the addition and multiplication, that are compatible with the one induced by the ring quotient, on the canonical representatives.

For instance:

$$\begin{aligned}(p_1 y + p_2)(q_1 y + q_2) &= p_1 q_1 y^2 + (p_1 q_2 + q_1 p_2) y + p_2 q_2 \\ &= (p_1 q_2 + q_1 p_2) y + (p_1 q_1 (x^3 + ax + b) + p_2 q_2)\end{aligned}$$

leads to:

```
Notation XPoly := 'X^3 + A *: 'X + B.
Definition dotp (p q : ecring) := p.2 * q.2 + (p.1 * q.1) * Xpoly.
Definition mul (p q : ecring) := [ecp p.1*q.2 + p.2*q.1 *Y + dotp p q].
```

where `.1` and `.2` resp. stand for the first and second projections.

The set $\mathbb{K}[\mathcal{E}]$, as a ring quotient by a prime ideal, is an integral domain. As such, we are able to equip the type `ecring` with an `integralDomain` structure, proving all the required axioms of the structure. We can then use the `fraction` [6] library to built the type `{fraction ecring}` representing $\mathbb{K}(\mathcal{E})$, the field of fractions over $\mathbb{K}[\mathcal{E}]$.

3.2 Order and evaluation of rational functions

In complex analysis, the *zeros* and *poles* of functions, and their *order* of vanishing are notions related to analytic functions and their Laurent expansion; while in abstract algebra, they refer to algebraic varieties and discrete valuation rings [9]. For our formalization, we follow the more elementary definitions given in [12]. More precisely, the evaluation of a function $f \in \mathbb{K}(\mathcal{E})$ at a point $P = (x_P, y_P) \in \mathcal{E}$ is defined as follows:

Definition 3. *A rational function $f \in \mathbb{K}(\mathcal{E})$ is said to be regular at $P = (x_P, y_P)$ if there exists a representative g/h of f such that $h(x_P, y_P) \neq 0$. If f is regular at P, the evaluation of f at P is the value $f(P) = \frac{g(x_P, y_P)}{h(x_P, y_P)}$, which is independent of the representative of f. If f is not regular at P, then P is called a pole of f and the evaluation of f at P is defined as $f(P) = \infty$.*

However, such a definition cannot be formalized as-is. Instead, we rely on the following extra notions allowing us to decompose any rational function in some canonical representative:

Definition 4. *A function $u \in \mathbb{K}(\mathcal{E})$ is called a uniformizer at $P \in \mathcal{E}(\mathbb{K})$ if i) $u(P) = 0$, and ii) every non-zero function $f \in \mathbb{K}(\mathcal{E})$ can be written in the form $f = u^v g$ with $g(P) \neq 0, \infty$ and $v \in \mathbb{Z}$.*

The exponent v is independent from the choice of the uniformizer and is called the order *of f at P, a quantity denoted by $\mathrm{ord}_f(P)$.*

Lemma 2. *There exists a uniformizer for every point on the curve.*

To get an intuition of the previous definitions, one can make a parallel with the notion of multiplicity for roots of univariate polynomials or with the notion of zeros and poles in $\mathbb{K}(x)$, the field of plain rational functions.

For instance, let us first consider the ring of polynomials $\mathbb{K}[x]$. Let p be a polynomial in $\mathbb{K}[x]$ and r be an element of $\mathbb{K}$. We can factorize p as $p = (x-r)^m q$ such that $m \in \mathbb{N}$ and $q(r) \neq 0$. The exponent m is the multiplicity of p at r. The multiplicity of r is 1 for the polynomial factor $(x-r)$. Evaluation and multiplicity are closely related: r is a root of p iff $m > 0$.

In an analogous way, we can consider the field of fractions $\mathbb{K}(\mathcal{E})$. Let P be in $\mathcal{E}$ and f in $\mathbb{K}(\mathcal{E})$. Then, one can always write f in the form $f = u^v g$ with $v \in \mathbb{Z}$ uniquely defined and P neither a zero nor a pole of g ($g(P) \neq 0, \infty$). The exponent v is the order of f at P. (Here, the function u corresponds to the polynomial factor $(x - r)$ for univariate polynomials) If $v > 0$ then P is a zero for f, and if $v < 0$ then P is a pole for f.

As said, the given definition of evaluation is not constructive. However, the proof of Lemma 2 is constructive and gives all the necessary material to define these notions. Let P be a point on the curve. For every $f \in \mathbb{K}[\mathcal{E}]$ (of type `ecring`) we explicitly give the decomposition $f = u_P^v(n/d)$ such that $n(P), d(P) \neq 0$, and u_P is a fixed rational function depending solely on P:

```
Definition unifun (P : point) : {fraction ecring} :=
  match P with
  | (x, y) => if y == 0 then [ecp 1 *Y + 0] else [ecp 0 *Y + ('X - x)]
  | EC_Inf => [ecp 0 *Y + X] / [ecp 1 *Y + 0]
  end.

Definition poly_order (f : ecring) (P : point) :=
  match P with
  | EC_Inf => let d := (degree f).-1 in
      (-d, ('X^d * f, [ecp 1 *Y + 0]^d)).
  | (x, y) => ...
```

and then prove that the decomposition is correct and unique:

```
Definition uniok (f u : fraction ecring) (p : point) o (n d : ecring) :=
  match p with
  | (x, y) => [&& f == u^o * (n // d), n.[x, y] != 0 & d.[x, y] != 0]
  | EC_Inf => ...

Lemma poly_order_correct:
  forall (f : ecring) (p : point), f != 0 -> oncurve p ->
    let: (o, (g1, g2)) := poly_order f p in
      uniok (unifun p) f p o g1 g2.

Lemma uniok_uniq:
  forall f p, f != 0 -> oncurve p ->
```

```
  forall o1 o2 n1 n2 d1 d2,
      uniok (unifun p) f p o1 n1 d1
    -> uniok (unifun p) f p o2 n2 d2
    -> (o1 == o2) && (n1 // d1 == n2 // d2).
```

We then lift these definitions to the quotient `{fraction ecring}`, and prove that all the lifted functions are stable by taking the quotient, allowing us to lift all the proved properties over $\mathbb{K}[\mathcal{E}]$ to $\mathbb{K}(\mathcal{E})$ (i.e. from `ecring` to `{fraction ecring}`). For instance, the order on `{fraction ecring}` is defined as:

```
Definition orderf (f : {ratio ecring}) p : int :=
  if \n_f == 0 then 0 else (poly_order \n_f p).1 - (poly_order \d_f p).1.

Definition order (f : {fraction ecring}) p := orderf (repr f) p.
```

We can then formalize Definition 3 by a simple case analysis over the *order*, relying on the decomposition of rational functions we have just formalized:

```
Definition eval (f : {fraction ecring}) p :=
  match p, order f p with
  | _, Posz _.+1 => 0
  | _, Negz _ => [inf]
  | (x, y), Posz 0 => (decomp f ecp).1.[x,y] / (decomp f ecp).2.[x,y]
  | EC_Inf, _ => ...
  end.
```

Due to the lack of space, we cannot give much details on the whole formalization of valuation theory, and move to the key lemma of this section:

Lemma 3. *A rational function $f \in \mathbb{K}(\mathcal{E})$ has a finite number of poles and zeros. Moreover, assuming that $\mathbb{K}$ is algebraically closed, $\sum_{P\in\mathcal{E}}(\mathrm{ord}_P(f)) = 0$.*

This lemma will be central when moving to the construction of the isomorphism between an elliptic curve and its Picard group.

3.3 Principal Divisors

From now on, we assume that $\mathbb{K}$ is algebraically closed.

Principal divisors are introduced as a tool for describing the zeros and poles of rational functions on an elliptic curve:

Definition 5 (Principal divisors). *Given $f \in \mathbb{K}(\mathcal{E})$, $f \neq 0$, the* principal divisor $\mathrm{Div}(f)$ *of f is defined as the formal (finite) sum:*

$$\mathrm{Div}(f) = \textstyle\sum_{P\in\mathcal{E}}(\mathrm{ord}_P(f))(P).$$

Note that $\mathrm{Div}(f)$ is well defined because a rational function has only finitely many zeros and poles. We write $\mathrm{Prin}(\mathcal{E})$ for the set of principal divisors.

The set $\mathrm{Prin}(\mathcal{E})$ forms a subgroup of $\mathrm{Div}(\mathcal{E})$, the set of formal sums over $\mathcal{E}$, a notion that we define now.

Definition 6. *A* divisor *on an elliptic curve $\mathcal{E}$ is a formal sum of points*

$$D = \sum_{P \in \mathcal{E}} n_P(P),$$

where $n_P \in \mathbb{Z}$, only finitely many nonzero. In other words, a divisor is any expression taken in the free abelian group generated over $\mathcal{E}(\mathbb{K})$. The domain *of D is $\mathrm{dom}(D) = \{P \mid n_P \neq 0\}$, and its* degree *is $\deg(D) = \sum_{P \in E} n_P$. For any point P, the* coefficient *of P in D is $\mathrm{coeff}(P, D) = n_P$.*

We write $\mathrm{Div}(\mathcal{E})$ for the set of divisors on $\mathcal{E}$, and $\mathrm{Div}^0(\mathcal{E})$ its subgroup composed of divisors of degree 0.

The set of divisors on $\mathcal{E}$ is an abelian group. The zero divisor is the unique divisor with all its coefficient set to 0, whereas the sum of two divisors is defined as the point-wise addition.

Based on the `quotient` libraries of SSReflect, we develop the theory of free abelian groups. Let `T` be a type. We first define the type of *pre-free group* as the collection of all sequences `s` of type `int * T` s.t. no pair of the form `(0, _)` can appear in `s` and for any `z : T`, a pair of the form `(_, z)` can appear at most once in `s`.

```
Definition reduced (D : seq (int * T)) :=
    (uniq [seq zx.2 | zx <- D])
  && (all [pred zx | zx.1 != 0] D).

Record prefreeg : Type := mkPrefreeg {
  seq_of_prefreeg : seq (int * T);
  _ : reduced seq_of_prefreeg
}.
```

The intent of `prefreeg` is to give a unique representation of a free-group expression, up to the order of the coefficients. For instance, if $D = k_1x_1 + \cdots + k_nx_n$ (with all the x_i's pairwise distinct and all the k_i's in $\mathbb{Z}^*$), then the `reduced` sequence `s = [:: (k_1, x_1), ..., (k_n, x_n)]`, or any sequence equal up to a permutation to `s`, is a valid representation of D. The type `freeg` of free-groups is then obtained by quotienting `prefreeg` by the `perm_eq` equivalence relation.

From there, we equip the type `freeg` with a group structure (the operation is noted additively), and define all the usual notions related to free groups (domain, coefficient, degree, ...). For instance, assume `G : zmodType` (G is a $\mathbb{Z}$-module) and `f : T -> G`. Then, `f` defines a unique group homomorphism from `freeg` to `G` that can be defined as follows:

```
Definition prelift (D : seq (int * T)) : G :=
  \sum_(x <- D) (f x.2) * x.1.
```

```
Definition lift (s : prefreeg T) : G := prelift s.
Definition fglift (D : {freeg T}) := lift (repr D).
```

One can check that the `fglift` function defines the homomorphism

$$\sum_{(z,x)\in D} z f(x)$$

The coefficient `coeff` and degree `deg` functions can be then defined as:

```
Definition coeff (t : T) (D : {freeg T}) :=
  fglift (fun x => (x == t)) D.

Definition deg (D : {freeg K}) : int :=
  fglift (fun x => 1) D.
```

The Group of Principal Divisors Returning to principal divisors, one can now check that $\mathrm{Prin}(\mathcal{E})$ is a subgroup of $\mathrm{Div}^0(\mathcal{E})$. Indeed, i) $\deg(divf) = 0$ by Lemma 3, and ii) since the order function is multiplicative ($\mathrm{ord}_p(f/g) = \mathrm{ord}_p(f) - \mathrm{ord}_p(g)$), we have $div(f/g) = div(f) - div(g)$.

Moreover, it is now clear that the coefficients associated in $\mathrm{Div}(f)$, to each point P, is the order of the function f at P, highlighting the fact that a divisor wraps up the zeros and poles of f.

Formally, we define principal divisors for polynomials on the curve with the function `ecdivp`:

```
Definition ecdivp (f : ecring) : {freeg (point)} :=
  \sum_(p <- ecroots f)
     << (order f (p.1, p.2)) * (p.1, p.2) >>
   + << order f EC_Inf * EC_Inf >>.
```

where `<< z * P >>` stands for the divisor $z(P)$ and the function `ecroots` takes a polynomial of $\mathbb{K}[\mathcal{E}]$ and returns the list of its finite zeros:

```
Definition ecroots f : seq (K * K) :=
  let forx := fun x =>
    let sqrts := roots ('X^2 - ('X^3 + A *: 'X + B).[x]) in
      [seq (x, y) | y <- sqrts & f.[x, y] == 0]
  in
    undup (flatten ([seq forx x | x <- roots (norm f)])).
```

The function `ecroots` relies on $\mathrm{norm}(f)$, a polynomial in $\mathbb{K}[x]$ associated to f that has the following property: (x, y) is a zero of f if and only if x is a zero of $\mathrm{norm}(f)$ and $y^2 = x^3 + ax + b$.

Next, we lift the definition of principal divisors to $\mathbb{K}(\mathcal{E})$, prove its correctness and recast the key Lemma 3 (`deg_ecdiv_eq0`):

```
Notation "\n_f" := (numerator f).
Notation "\d_f" := (denominator f).

Definition ecdiv_r (f : {ratio ecring}) :=
  if \n_f == 0 then 0 else (ecdivp \n_f) - (ecdivp \d_f).

Definition ecdiv := lift_fun1 {fraction ecring} ecdiv_r.

Lemma ecdiv_coeffE (f : {fraction ecring}) p:
  coeff p (ecdiv f) = order f p.

Lemma deg_ecdiv_eq0 (f : {fraction ecring}): deg (ecdiv f) = 0.
```

3.4 Divisor of a line

Before moving to the definition of the Picard group, we characterize the divisors of some specific rational functions. These divisors will later help formalize the construction of the Picard group:

Definition 7. *A line $l \in \mathbb{K}(\mathcal{E})$ is any rational function of the form $l(x, y) = ax + by + c$ with $a, b, c \in \mathbb{K}$ not all zero.*

For instance, if (PQ) is the line intersecting the curve at P and Q, then we know that (PQ) intersects $\mathcal{E}$ at exactly three points (counting multiplicities): P, Q and $P \boxdot Q$. Assuming that P, Q and $P \boxdot Q$ are all finite, these three points are the unique zeros of the rational function l associated to (PQ) and $\mathrm{Div}(l) = (P)+(Q)+(P\boxdot Q)-3(\mathcal{O})$. This relation still holds when one or several of these three points are equal to $\mathcal{O}$. For instance, $\mathrm{Div}(x - x_P) = (P)+(-P)-2(\mathcal{O})$, where $x - x_P$ is the line intersecting $\mathcal{E}$ at P, $-P$ and $\mathcal{O}$.

3.5 The Picard Group

Definition 8. *The Picard group* $\mathrm{Pic}(\mathcal{E})$ *is the group quotient* $\mathrm{Div}^0(\mathcal{E})/\mathrm{Prin}(\mathcal{E})$. *Note that the degree is well defined on the divisor class group since if* $D_1 = D_2 + \mathrm{Div}(f)$ *then* $\deg D_1 = \deg D_2 + \deg(\mathrm{Div}(f)) = \deg D_2 + 0 = \deg D_2$.

In other words, $\mathrm{Pic}(\mathcal{E})$ is defined as the quotient of $\mathrm{Div}^0(\mathcal{E})$ by the following equivalence relation $\sim$:

$$D_1 \sim D_2 \text{ if and only if } \exists f \in \mathbb{K}(\mathcal{E}) \text{ such that } div(f) = D_1 - D_2.$$

a notion that we formalize as follows:

```
Definition ecdeqv D1 D2 :=
  (exists f : {fraction ecring}, ecdiv f = D1 - D2).
Notation "D1 :~: D2" := (ecdeqv D1 D2).
```

We do not give a direct construction of $\mathrm{Pic}(\mathcal{E})$ but instead prove that any class of $\mathrm{Pic}(\mathcal{E})$ can be represented by a divisor of the form $(P) - (\mathcal{O})$.

The construction of this representative is based on a procedure called *Linear Reduction*. Assume that P and Q are two finite points of $\mathcal{E}(\mathbb{K})$. We know that the divisor of the line l intersecting $\mathcal{E}$ at P and Q is $\mathrm{Div}(l) = (P) + (Q) + (P \boxdot Q) - 3(\mathcal{O})$. Likewise, the divisor of the line l' intersecting $\mathcal{E}$ at $P \boxdot Q$ and $-(P \boxdot Q)$ $(= P + Q)$ is $\mathrm{Div}(l') = (P + Q) + (P \boxdot Q) - 2(\mathcal{O})$. Hence,

$$\begin{aligned}\mathrm{Div}(l/l') &= \mathrm{Div}(l) - \mathrm{Div}(l') \\ &= (P) + (Q) - (P + Q) - (\mathcal{O})\end{aligned}$$

and, $(P) + (Q) \sim (P + Q) + (\mathcal{O})$.

Iterating this procedure, we can reduce any divisor of the form:

$$(P_1) + \cdots + (P_n) - (Q_1) - \cdots - (Q_k) + r(\mathcal{O})$$

to an equivalent one $(P) - (Q) + r'(\mathcal{O})$, with $r' \in \mathbb{Z}$. Using one more time the same construction, one can show that $(P) - (Q) + n'(\mathcal{O})$ is equivalent to $(P - Q) + n''(\mathcal{O})$ where $n', n'' \in \mathbb{Z}$.

The `lr` function formally defines the linear reduction procedure:

```
Definition fgpos (D : {freeg K}) :=
  \sum_(p <- dom D | coeff p D > 0) coeff p D.
Definition fgneg (D : {freeg K}) :=
  \sum_(p <- dom D | coeff p D < 0) -(coeff p D).

Definition lr_r (D : {freeg point}) :=
  let iter p n := iterop _ n + p EC_Inf in
    \sum_(p <- dom D | p != EC_Inf) (iter p `|coeff p D|).

Definition lr (D : {freeg point}) : point :=
  let: (Dp, Dn) := (fgpos D, fgneg D) in
    lr_r Dp - lr_r Dn.

Lemma ecdeqv_lr D: all oncurve (dom D) ->
  D :~: << lr D >> + << deg D - 1 *g EC_Inf >>.
```

where `(Dp, Dn) := (fgpos D, fgneg D)` is the decomposition of D into its negative and positive parts.

The lemma `ecdeqv_lr` states that any divisor is equivalent to a divisor of the form $(P) + (\deg D - 1)(\mathcal{O})$. In the context of $\mathrm{Pic}(\mathcal{E})$, this means that any class contains a divisor of the form $(P) - (\mathcal{O})$ (recall that $\mathrm{Pic}(\mathcal{E})$ is a group quotient of $\mathrm{Div}^0(\mathcal{E})$ — the divisors of degree 0). In the next section, we end the construction of the Picard group by proving that at most one such representative can be found in each class of $\mathrm{Pic}(\mathcal{E})$.

4 Linking Pic($\mathcal{E}$) to $\mathcal{E}(\mathbb{K})$

In this section, we finish our formal construction of the Picard group and prove the existence of an isomorphism between Pic($\mathcal{E}$) and $\mathcal{E}(\mathbb{K})$. We start by defining a canonical representative for the classes of Pic($\mathcal{E}$):

Lemma 4. *For every class of* Pic($\mathcal{E}$)*, there exists a unique representative of the form* $(P) - (\mathcal{O})$ *with* $P \in \mathcal{E}(\mathbb{K})$*.*

From Section 3.5, we already know that each class of Pic($\mathcal{E}$) contains one such representative. Assume now that $(P)-(\mathcal{O})$ and $(Q)-(\mathcal{O})$ are two representatives of a class of Pic($\mathcal{E}$) with $P \neq Q$. Such an assumption allows us to find a rational function $h \in \mathbb{K}(\mathcal{E})$ s.t. every rational function $f \in \mathbb{K}(\mathcal{E})$ can be expressed as a polynomial fraction of h. This implies that $K(\mathcal{E})$ and $K(x)$ are isomorphic. However, since $K(\mathcal{E})$ is a field extension of degree 2 of $K(x)$, such an h cannot exist. Hence, $P = Q$:

```
Lemma lr_uniq: << p >> :~: << q >> -> p = q.
```

The Picard group can now be formally defined as the set of divisors of the form $(P) - (\mathcal{O})$. It remains to prove the existence of a bijection between Pic($\mathcal{E}$) and $\mathcal{E}(\mathbb{K})$. Namely, the function

$$\begin{aligned} \phi : \mathcal{E}(\mathbb{K}) &\to \mathrm{Pic}(\mathcal{E}) \\ P &\mapsto [(P) - (\mathcal{O})] \end{aligned}$$

is our isomorphism. Indeed, ϕ is clearly bijective and from the results of Section 3:

$$\begin{aligned} \phi(P_1) - \phi(P_2) &= [(P_1) - (\mathcal{O})] - [(P_2) - (\mathcal{O})] = [(P_1) - (P_2)] \\ &= [(P_1 - P_2) - (\mathcal{O})] = \phi(P_1 - P_2). \end{aligned}$$

In our formalization, we directly use the linear reduction function `lr` in place of ϕ^{-1}. For instance, we prove that `lr` commutes with the curve operations and maps $(P) - (\mathcal{O})$ to $P \in \mathbb{K}(\mathcal{E})$:

```
Lemma lrB: forall (D1 D2: {freeg point},
   deg D1 = 0 -> all oncurve (dom D1) ->
   deg D2 = 0 -> all oncurve (dom D2) ->
   lr (D1 - D2) = lr D1 - lr D2.

Lemma lrpi: forall p : point,
  oncurve p -> lr (<<p>> - <<EC_Inf>>) = p.
```

This allows us to transport the structure from Pic($\mathcal{E}$) to $\mathcal{E}(\mathbb{K})$, proving that $\mathcal{E}(\mathbb{K})$ is a group.

5 Related Work

Hurd et al. [13] formalize elliptic curves in higher order logic using the HOL-4 proof assistant. Their goal is to create a "gold-standard" set of elliptic curve operations mechanized in HOL-4, which can be used afterwards to verify ec-algorithms for scalar multiplication. They define datatypes to represent elliptic curves on arbitrary fields (in both projective and affine representation), rational points and the elliptic curve group operation, although they do not provide a proof that the operation indeed satisfies the group properties. In the end, they state the theorem that expresses the functional correctness of the ElGamal encryption scheme for elliptic curves.

Smith et al. [18] use the Verifun proof assistant to prove that two representations of an elliptic curve in different coordinate systems are isomorphic. Their theory applies to elliptic curves on prime fields. They define data structures for affine and projective points and the functions that compute the elliptic curve operations in affine and Jacobian coordinates. In their formalization there is no datatype for elliptic curves, an elliptic curve is a set of points that satisfy a set of conditions. They define the transformation functions between the two systems of coordinate and prove that for elliptic curve points the transformation functions commute with the operations and that both representations of elliptic curves in affine or Jacobian coordinates are isomorphic.

Théry [20] present a formal proof that an elliptic curve is a group using the Coq proof assistant. The proof that the operation is associative relies heavily on case analysis and requires handling of elementary but subtle geometric transformations and therefore uses computer-algebra systems to deal with non-trivial computation. In our development, we give a different proof of the associativity of the elliptic curve group law: we define an algebraic structure (the Picard group of divisors) and proceed to prove that the elliptic curve is isomorphic to this structure. Our formalization is more structural than [20] in the sense that it involves less computation and the definition of new algebraic structures.

As in [13] and [18] we wish to develop libraries that will enable the formal analysis of elliptic curve algorithms and our proofs follow textbook mathematics. As in [20], we give a formal proof of the group law for elliptic curves. Nevertheless, the content of our development is quite different from the related work. To the extent of our knowledge this is the first formalization of divisors and rational functions of a curve, which are objects of study of algebraic geometry. Such libraries may allow the formalization of non-trivial algorithms that involve divisors (such as the Miller algorithm for pairings [16]), isogenies (such as [3], [8]) or endomorphisms on elliptic curves (such as the GLV algorithm for scalar multiplication [10]).

6 Future work

This paper presents a formalization of the elementary elliptic curve theory in the SSReflect extension of Coq. Our central result is the formal proof of the Picard

theorem which is a structure theorem for elliptic curves. A direct implication of this theorem is the associativity of the elliptic curve group operation. Our development includes generic libraries formalizing divisors and rational functions that are designed to enable the formal verification of elliptic curve cryptographic algorithms. Our formalization required 10k lines of code out of which 6.5k lines were required for the proof of the Picard theorem. The SSReflect features and methodology for the formalization of algebraic structures have been very helpful to our development.

The proof layout follows the ones that can be found in most graduate text books about smooth algebraic curves, but instantiated to the case of elliptic curves. Generalizing our development should not deeply change the general structure, but will certainly require the development of a lot of background theory: affine spaces, multinomials, rational maps, ring quotients, valuation rings, formal differentials, ... to name some of them.

To further validate our development, we are working on the formal proof of correctness of an implementation of the GLV algorithm [10]. The GLV algorithm is a non-generic scalar multiplication algorithm that uses endomorphisms on elliptic curves to accelerate computation. It is composed of three independent algorithms: parallel exponentiation, decomposition of the scalar, computation of endomorphisms on elliptic curves. The third algorithm involves background from algebraic geometry that can be provided by the rational functions' libraries presented in this paper. We aim to generate a certified implementation of the GLV algorithm based on the methodology described in [7]. Another algorithm that would be interesting to formalize is the Miller algorithm for bilinear pairings [16], which would rely on our divisors' library to compute pairings by evaluating linear functions on divisors.

In the development presented in this paper, we chose to represent elliptic curves in an affine coordinate system. However, projective, Jacobian and other coordinate systems are widely used in practice mainly for reasons of efficiency. Indeed, nowadays the search of optimal coordinate-systems is an active domain of research in elliptic curve cryptography. Future work is to extend our libraries to isomorphic coordinate representations in order to allow formal analysis of algorithms in different coordinate systems. Furthermore, in our development we treat elliptic curves on fields with characteristic different from 2 and 3 although in cryptography binary fields are used often. Generalizing our development to more general curves is a natural extension.

Acknowledgements

We thank Philippe Guillot and Benjamin Smith for their discussions on the theory of elliptic curves. We also thank Cyril Cohen, Assia Mahboubi and all the SSReflect experts for spending time explaining the internals of the SSReflect libraries. Finally, we thank Karthikeyan Bhargavan, Cătălin Hriţcu, Robert Kunnemann, Alfredo Pironti, Ben Smyth for their feedback and discussions on this paper.

References

1. The mathematical components project. Online – http://www.msr-inria.fr/projects/mathematical-components/.
2. Matteo Avalle, Alfredo Pironti, and Riccardo Sisto. Formal verification of security protocol implementations: a survey. *Formal Aspects of Computing*, pages 1–25, 2012.
3. Eric Brier and Marc Joye. Fast point multiplication on elliptic curves through isogenies. In *AAECC*, pages 43–50, 2003.
4. B.B. Brumley, M. Barbosa, D. Page, and F. Vercauteren. Practical realisation and elimination of an ecc-related software bug attack. Cryptology ePrint Archive, Report 2011/633, 2011. http://eprint.iacr.org/.
5. L.S. Charlap and D.P. Robbins. Crd expository report 31 an elementary introduction to elliptic curves, 1988.
6. Cyril Cohen. Pragmatic quotient types in coq. In *ITP*, pages 213–228, 2013.
7. Maxime Dénès, Anders Mörtberg, and Vincent Siles. A refinement-based approach to computational algebra in coq. In *ITP*, pages 83–98, 2012.
8. Christophe Doche, Thomas Icart, and David R. Kohel. Efficient scalar multiplication by isogeny decompositions. In *Public Key Cryptography*, pages 191–206, 2006.
9. William Fulton. *Algebraic curves - an introduction to algebraic geometry (reprint vrom 1969)*. Advanced book classics. Addison-Wesley, 1989.
10. Robert P. Gallant, Robert J. Lambert, and Scott A. Vanstone. Faster point multiplication on elliptic curves with efficient endomorphisms. In *CRYPTO*, pages 190–200, 2001.
11. Georges Gonthier, Assia Mahboubi, and Enrico Tassi. A Small Scale Reflection Extension for the Coq system. Research Report RR-6455, INRIA, 2008.
12. Philippe Guillot. *Courbes Elliptiques, une présentation élémentaire pour la cryptographie*. Lavoisier, 2010.
13. Joe Hurd, Mike Gordon, and Anthony Fox. Formalized elliptic curve cryptography. High Confidence Software and Systems, 2006.
14. Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, January 1987.
15. Victor S. Miller. Use of elliptic curves in cryptography. In *CRYPTO*, pages 417–426, 1985.
16. Victor S. Miller. Short programs for functions on curves. In *IBM THOMAS J. WATSON RESEARCH CENTER*, 1986.
17. Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.
18. Eric Whitman Smith and David L. Dill. Automatic formal verification of block cipher implementations. In *FMCAD*, pages 1–7, 2008.
19. The Coq development team. The Coq Proof Assistant Reference Manual Version 8.4. Online – http://coq.inria.fr, 2013.
20. Laurent Théry. Proving the group law for elliptic curves formally. Technical Report RT-0330, INRIA, 2007.