COUNTING POINTS ON ELLIPTIC CURVES

OLE ANDRE BIRKEDAL

Abstract. hei hei

1. Schoof's algorithm

1.1. **Division polynomials.** Recall for this section that an elliptic curve corresponds to a lattice Λ so we have an isomorphism

$$\bar{k}/\Lambda \simeq E(\bar{k})$$
$$z \mapsto (\wp(z), \wp'(z))$$

where $\wp(z)$ is the elliptic Weierstrass function.

Definition 1. *The* division polynomials *are polynomials* $\Psi_n(x,y) \in \mathbb{Z}[x,y,A,B]$ *defined by the recurrence relations*

$$\begin{aligned}\Psi_0 &= 0\\ \Psi_1 &= 1\\ \Psi_2 &= 2y\\ \Psi_3 &= 3x^4 + 6Ax^2 + 12Bx - A^2\\ \Psi_{2n+1} &= \Psi_{n+2}\Psi_n^3 - \Psi_{n+1}^3\Psi_{n-1}\\ \Psi_{2n} &= (2y)^{-1}\Psi_n(\Psi_{n+2}\Psi_{n-1}^2 - \Psi_{n-2}\Psi_{n+1}^2)\end{aligned}$$

where $\Psi_n(x,y) = 0$ *is and only if* $(x,y) \in E[n]$.

The construction of these polynomials can be done in at least two ways and I will discuss both of them briefly.

One way of doing this is to construct a function having poles at the n-torsion points of our elliptic curve as follows

$$f_n(z) = n^2 \prod(\wp(z) - \wp(u))$$

where the product is taken over all n-torsion points of $\bar{k}/\Lambda$, denoted $\bar{k}/\Lambda[n]$. This function has roots at exactly the n-torsion points by definition, which is at least what we want. A more throrough examination of this method can be found in [serge lang-ref].

Another way which is more elementary by highly computational is to work explicitly with the addition formulas for elliptic curves. + mer forklaring.

Replacing the terms y^2 in Ψ_n by $x^3 + Ax + B$ we obtain polynomials Ψ'_n in $\mathbb{F}_q[x]$ if is n is odd or $y\mathbb{F}_q[x]$ if n is even. To avoid this distinction we define

$$f_n(x) = \begin{cases}\Psi'_n(x,y) & \text{if n is odd}\\ \Psi'_n(x,y)/y & \text{if n is even}\end{cases}$$

Proposition 1. *Let* $n \geq 2$ *and* Ψ_n *the division polynomial as defined above, then*

$$nP = (x - \frac{\Psi_{n-1}\Psi_{n+1}}{\Psi_n^2}, \frac{\Psi_{n+2}\Psi_{n-1}^2 - \Psi_{n-2}\Psi_{n+1}^2}{4y\Psi_n^3})$$

1.2. **Computing the number of points.** For an elliptic curve over $\mathbb{F}_q$ given by

$$E : y^2 = x^3 + Ax + B$$

we want to compute the size of $\#E(\mathbb{F}_q)$, we know from before that

$$\#E(\mathbb{F}_q) = q + 1 - t$$

where t is the trace of the Frobenius as seen in section [referanse]. We know that t satisfies the Hasse bound namely

$$|\#E(\mathbb{F}_q) - q - 1| = |t| < 2\sqrt{q}$$

Let $S = \{3, 5, 7, 11, \dots L\}$ be the set of odd primes $\leq L$ such that the product is bigger than the Hasse interval

$$N = \prod_{\ell \in S} \ell > 4\sqrt{q}$$

If we can then calculate $t\,(mod \ell)$ for all $\ell \in S$ we can uniquely determine $t\,(mod N)$ by invoking the Chinese remainder theorem, which then by the Hasse bound is our Frobenius trace t.

The argument above is the gist of Schoof's algorithm, we will now look at how to calculate $t\,(mod \ell)$. Let ϕ be the Frobenius endomorphism resticted to $E[\ell]$ and let q_ℓ, τ be q and t reduced modulo ℓ respectively. The computation of τ can then be done by checking if

$$\phi^2(P) + q_\ell P = \tau \phi(P)$$

holds for $P \in E[\ell]$. To perform the addition on the left hand side of the equality we need to distinguish the cases where the points are on a vertical line or not. In other words we have to verify if for $P = (x, y) \in E[\ell]$ the following holds

$$\phi^2(P) = \pm q_\ell P$$

Noting that $-P = (x, -y)$ we write out the equality for the x-coordinates in terms of division polynomials

$$x^{q^2} = x - \frac{\Psi_{q_\ell - 1}\Psi_{q_\ell + 1}}{\Psi_{q_\ell}^2}(x, y)$$

Writing this out in terms of $f_n(x)$ and noting that for n even we have $\Psi_n(x, y) = y f_n(x)$, a calculation for q_ℓ even yields

$$\begin{aligned} x^{q^2} &= \frac{f_{q_\ell - 1}(x) f_{q_\ell + 1}(x)}{(f_{q_\ell} y)^2} \\ &= \frac{f_{q_\ell - 1}(x) f_{q_\ell + 1}(x)}{f_{q_\ell}^2 (x^3 + Ax + B)} \end{aligned}$$

The calculation for q_ℓ odd is similar and we get the equality

$$x^{q^2} = \begin{cases} x - \frac{f_{q_\ell - 1}(x) f_{q_\ell + 1}(x)}{f_{q_\ell}^2 (x^3 + Ax + B)} & \text{if } q_\ell \text{ is even} \\ x - \frac{f_{q_\ell - 1}(x) f_{q_\ell + 1}(x)(x^3 + Ax + B)}{f_{q_\ell}^2 (x)} & \text{if } q_\ell \text{ is odd} \end{cases}$$

We thus get two equations and we want to verify they have any solutions $P \in E[\ell]$. For doing this we compute the following greatest common divisors

$$gcd((x^{q^2} - x) f_{q_\ell}^2 (x^3 + Ax + B) + f_{q_\ell - 1}(x) f_{q_\ell + 1}(x), f_\ell(x)) \quad (q_\ell \text{ even})$$

$$gcd((x^{q^2} - x) f_{q_\ell}^2 (x) + f_{q_\ell - 1}(x) f_{q_\ell + 1}(x)(x^3 + Ax + B), f_\ell(x)) \quad (q_\ell \text{ odd})$$

We are now going to treat the rest in two cases, depending on the value from the above gcds.

Case 1: $gcd \neq 1$ meaning there exist a non-zero ℓ-torsion point P such that $\phi^2(P) = \pm q_\ell P$. If $\phi^2(P) = -q_\ell P$ we have that $\tau\phi(P) = 0$ but since $\phi(P) \neq 0$ we know that $\tau = 0$. If $\phi^2(P) = q_\ell P$ we have that

$$2q_\ell P = \tau\phi(P) \Leftrightarrow \phi(P) = \frac{2q_\ell}{\tau}$$

Substituting the last equality into $\phi^2(P) = q_\ell P$ we obtain

$$\frac{4q_\ell^2}{\tau^2} = q_\ell P \Leftrightarrow 4q_\ell P = \tau^2 P$$

We thus obtain the congruence $\tau^2 \equiv 4q_\ell\, (mod \ell)$

Case 2: $gcd = 1$ so $\phi^2(P) \neq \pm q_\ell P$ meaning the two points are not equal nor are they on the same vertical line for any ℓ-torsion point P. This enables us to do the addition $\phi^2(P) + q_\ell P$ using the appropriate addition formulas. Recall that if $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ are two points on E with $P \neq Q$ we have that their sum is given by $P + Q = (x_3, y_3)$ where

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$
$$x_3 = -x_1 - x_2 + \lambda^2$$
$$y_3 = -y_1 - \lambda(x_3 - x_1)$$

We can now write out the addition explicitly in terms of polynomials as follows

$$\lambda = \frac{\Psi_{q_\ell+2}\Psi_{q_\ell-1}^2 - \Psi_{q_\ell-2}\Psi_{q_\ell+1}^2 - 4y^{q^2+1}\Psi_{q_\ell}^3}{4\Psi_{q_\ell}y((x - x^{q^2})\Psi_{q_\ell}^2 - \Psi_{q_\ell-1}\Psi_{q_\ell+1}}$$

1.3. Modular polynomials.

fixme

2. Satoh's algorithm

fixme