Small Scalar Multiplication on Weierstrass Curves using Division Polynomials

Sergey Agievich, Stanislav Poruchnik, and Vladislav Semenov
Research Institute for Applied Problems of Mathematics and Informatics
Belarusian State University

June 04, 2021 [Dorokhovo, Moscow region]

Table of contents

Weierstrass curves and scalar multiplication

Weierstrass curves

(short) Weierstrass curves:

$$E\colon y^2 = x^3 + ax + b, \quad a, b \in \mathbb{F}, \quad 4a^3 + 27b^2 \neq 0,$$

where $\mathbb{F}$ is a large prime finite field.

$\mathbb{F}$-rational points:

$$E(\mathbb{F}) = \{\text{affine points } (x, y) \in \mathbb{F}^2\} \cup \{\text{a special point } O\}.$$

Point addition: $+\colon E(\mathbb{F}) \times E(\mathbb{F}) \to E(\mathbb{F})$:

- defined by an arithmetic circuit over $\mathbb{F}$;
- a commutative group operation (O is zero);
- negation is not expensive.

Usage in cryptography

A group: $G \in E(\mathbb{F}) \setminus \{O\} \mapsto \mathbb{G} = \langle G \rangle$:

- the order $q = |\mathbb{G}|$ is a large prime;
- the cofactor $|E(\mathbb{F})|/q$ is small (for Weierstrass curves, it can achieve 1 that is convenient for various cryptographic protocols);
- $\mathbb{G}^* = \mathbb{G} \setminus \{O\}$.

The key observation (Miller, 1986; Koblitz, 1987). The DL (Discrete Logarithm) problem in $\mathbb{G}$ seems to be hard and, therefore, $\mathbb{G}$ can be used in cryptography.

Scalar multiplication

Fixed base settings: $d \mapsto dG$, $d \in \{1, 2, \ldots, q-1\}$.

[KeyGen, ElGamal.Sign, Schnorr.Sign]

Scalar multiplication

Fixed base settings: $d \mapsto dG$, $d \in \{1, 2, \ldots, q-1\}$.

[KeyGen, ElGamal.Sign, Schnorr.Sign]

Variable base settings: $(d, P) \mapsto dP$, $P \in \mathbb{G}^*$.

[Diffie-Hellman, ElGamal.Verify, Schnorr.Verify]

Scalar multiplication

Variable base settings: $(d, P) \mapsto dP$, $P \in \mathbb{G}^*$.

[Diffie-Hellman, ElGamal.Verify, Schnorr.Verify]

Arithmetic circuits for scalar multiplication

Circuit$[d]$

Input: coordinates of P, field constants.

Output: coordinates of dP.

Operations: $\mathtt{I}$ (inversion), $\mathtt{M}$ (multiplication), $\mathtt{S}$ (squaring), $\mathtt{A}$ (addition or subtraction), $\mathtt{m}$ (multiplication by a small constant); $\mathtt{h}$ (division by 2).

Requirements:

- the complexity of **Circuit**$[d]$ should be as small as possible (efficiency);
- the structure and complexity of **Circuit**$[d]$ must not depend on d (the constant-time property, security).

Heuristics for the complexity:
$h = \left(\frac{\mathtt{I}}{\mathtt{M}}, \frac{\mathtt{S}}{\mathtt{M}}, \frac{\mathtt{A}\approx\mathtt{m}\approx\mathtt{h}}{\mathtt{M}}\right) \in [80; 100] \times [0.6; 1.0] \times [0; 0.1]$.

$\mathtt{M}$-**complexity,** $\mathtt{M}(h)$: express the complexity as the number of $\mathtt{M}$ operations provided that a heuristic h is used.

Projective coordinates

Jacobian coordinates: $(X, Y, Z) \sim (x, y) = (X/Z^2, Y/Z^3)$.

Homogeneous coordinates: $(X, Y, Z) \sim (x, y) = (X/Z, Y/Z)$.

The coordinate Z acts as a normalizing factor that "absorbs" the expensive operation $\mathtt{I}$.

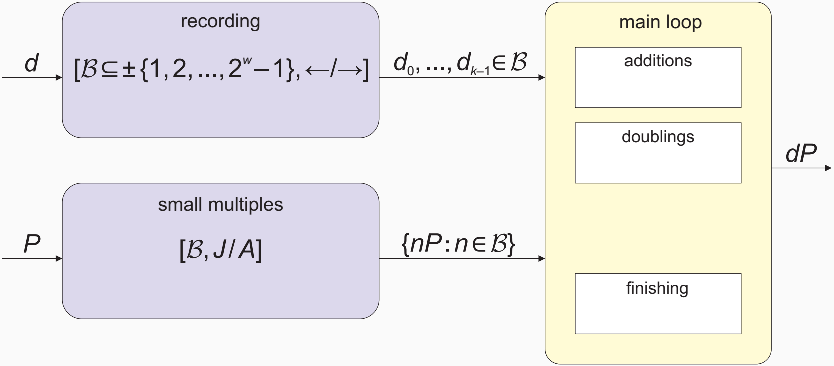
operation ($a = -3$)	multiplicative complexity	notes
$A \leftarrow A + A$	$1\mathtt{I} + 2\mathtt{M} + 1\mathtt{S}$	chord
$A \leftarrow 2A$	$1\mathtt{I} + 2\mathtt{M} + 2\mathtt{S}$	tangent
$J \leftarrow J + J$	$11\mathtt{M} + 5\mathtt{S}$	EFD[add-2007-bl]
$J \leftarrow J + A$	$7\mathtt{M} + 4\mathtt{S}$	EFD[dbl-1998-hnm]
$J \leftarrow 2J$	$3\mathtt{M} + 5\mathtt{S}$	EFD[madd-2007-bl]
$H \leftarrow H + H$	$14\mathtt{M}$	[RCB16], complete
$H \leftarrow H + A$	$13\mathtt{M}$	[RCB16], complete

complete formulas = both addition and doubling

EFD = Elliptic Formula Database (D. Bernstein, T. Lange)

RCB = J. Renes, C. Costello, L. Batina

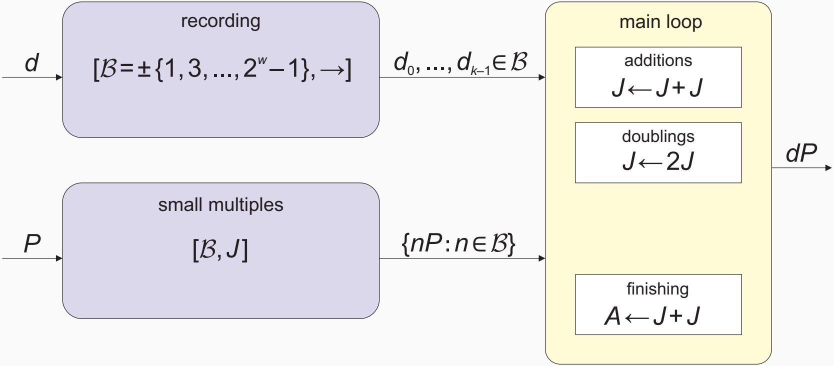
Window methods



Options:

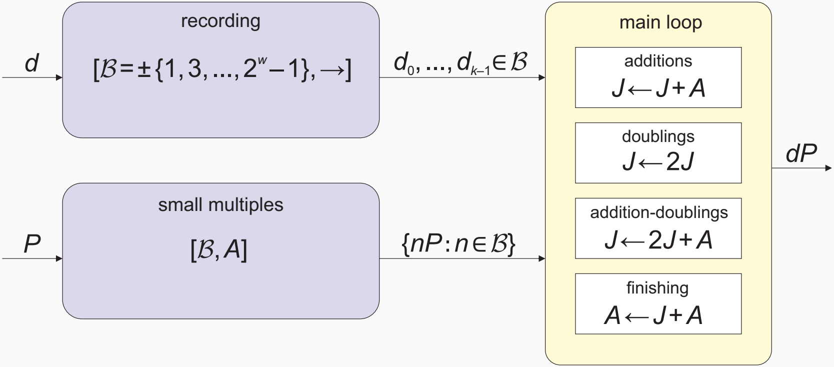
- w (window width);
- $\mathcal{B}$ (base);
- a direction of recording (recoding) the bits of d: $\rightarrow$ or $\leftarrow$;
- coordinates of small multiples: Jacobian or affine.

Strategy I



d
recording
$[\mathcal{B} = \pm\{1, 3, \ldots, 2^w - 1\}, \rightarrow]$
$d_0, \ldots, d_{k-1} \in \mathcal{B}$
P
small multiples
$[\mathcal{B}, J]$
$\{nP : n \in \mathcal{B}\}$
main loop
additions
$J \leftarrow J + J$
doublings
$J \leftarrow 2J$
finishing
$A \leftarrow J + J$
dP

Strategy II


recording
$[\mathcal{B} = \pm\{1, 3, \ldots, 2^w - 1\}, \rightarrow]$
d
$d_0, \ldots, d_{k-1} \in \mathcal{B}$
small multiples
$[\mathcal{B}, A]$
P
$\{nP : n \in \mathcal{B}\}$
main loop
additions
$J \leftarrow J + A$
doublings
$J \leftarrow 2J$
addition-doublings
$J \leftarrow 2J + A$
finishing
$A \leftarrow J + A$
dP

Division polynomials and small scalar multiplication

Division polynomials

If $P = (x, y)$ is an affine point, $n \geq 2$ and $nP \neq O$, then

$$nP = \left(x - \frac{\psi_{n-1}(P)\psi_{n+1}(P)}{\psi_n(P)^2}, \frac{\psi_{n+2}(P)\psi_{n-1}(P)^2 - \psi_{n-2}(P)\psi_{n+1}(P)^2}{4y\psi_n(P)^3} \right).$$

Here $\{\psi_n(P) = \psi_n(x, y)\}$ is **division polynomials**.

Related concepts: elliptic divisibility sequences (Ward 1947), elliptic nets (Stange, 2007).

Usage for scalar multiplication:

- V. Miller (1986): "we may calculate the coordinates of the above points in $26 \log_2 n$ multiplication";
- N. Kanayama et al. (2014) based on Stange (2007): double-and-add, $26\mathtt{M} + 6\mathtt{S}$ per bit (impractical);
- R. Chen et al. (2015): $22\mathtt{M} + 6\mathtt{S}$ per bit (impractical);
- our approach: use division polynomials only for small scalar multiplication.

Recursion

Auxiliary polynomials $W_n(x, y^2)$:

$$\psi_n(x, y) = \begin{cases} 2yW_n(x, y^2), & n \text{ is even}, \\ W_n(x, y^2), & n \text{ is odd}. \end{cases}$$

Assuming (x, y) is fixed, denote $W_n = W_n(x, y^2)$. With that,

$$W_n = \begin{cases} -1, & n = -1, \\ 0, & n = 0, \\ 1, & n = 1, \\ 1, & n = 2, \\ 3(x^2 + a)^2 + 4(3bx - a^2), & n = 3, \\ 2(x^4(x^2 + 5a) + bx(20x^2 - 4a) - 5a^2x^2 - 8b^2 - a^3), & n = 4, \\ W_m \left(W_{m+2} W_{m-1}^2 - W_{m-2} W_{m+1}^2\right), & n = 2m, \\ \left((2y)^4 W_m W_{m+2}\right) W_m^2 - \left(W_{m-1} W_{m+1}\right) W_{m+1}^2, & n = 4k + 1, \quad m = 2k, \\ \left(W_m W_{m+2}\right) W_m^2 - \left((2y)^4 W_{m-1} W_{m+1}\right) W_{m+1}^2, & n = 4k + 3, \quad m = 2k + 1. \end{cases}$$

`SmallMultJ`: Jacobian small multiples (Strategy I)

For $n = 3, 5, \ldots, 2^w - 1$:

$$\begin{aligned} nP &= (X_n, Y_n, W_n), \\ X_n &= xW_n^2 - (2y)^2 W_{n-1} W_{n+1}, \\ Y_n &= y\left(W_{n+2} W_{n-1}^2 - W_{n-2} W_{n+1}^2\right). \end{aligned}$$

Complexity:
$\approx 9.5\mathtt{M} + 3.5\mathtt{S}$ per point.

Naive approach (repeated additions with $2P$):
$\approx 11\mathtt{M} + 5\mathtt{S}$ per point.

SmallMultA: affine small multiples (Strategy II)

For $n = 3, 5, \ldots, 2^w - 1$:

$$nP = \left(\frac{X_n}{W_n^2}, \frac{Y'_n}{W_n^4}\right),$$
$$X_n = xW_n^2 - (2y)^2 W_{n-1} W_{n+1},$$
$$Y'_n = y\left(W_n W_{n+2} W_{n-1}^2 - W_{n-2} W_n W_{n+1}^2\right).$$

Montgomery's trick:

- simultaneous inversion of $\{W_n^2\}$;
- 1I: actually inverse only $\prod_n W_n^2$.

Integration

Recording

1. Write d in base 2^w:

$$d = \sum_{i=0}^{k-1} d_i 2^{wi}, \quad k = \lceil \mathrm{bitlen}(q)/w \rceil, \quad d_i \in \{0, 1, \ldots, 2^w - 1\}. \qquad (\star)$$

2 (d is odd). For $i = k-1, k-2, \ldots, 1$:

$$(d_i, d_{i-1}) \leftarrow (d_i + \mathtt{even}(d_i), d_{i-1} - \mathtt{even}(d_i) 2^w).$$

After that $(\star)$ is preserved but $d_i \in \pm\{1, 3, \ldots, 2^w - 1\}$.

3. If d is even, then

- $d \leftarrow q - d$;
- compute the point $(q - d)P = -dP$;
- negate it.

ScalarMult

Algorithm `ScalarMult`

Input: $P \in \mathbb{G}^*$, $d \in \{1, 2, \ldots, q-1\}$.
Output: dP (in affine coordinates).
Steps:

1. $\delta \leftarrow d \bmod 2$, $d \leftarrow (1-\delta)q + (2\delta - 1)d$.
2. Choose a window width w $(3 \leq w < \log_2 q)$.
3. $P[1] \leftarrow P$, $(P[3], P[5], \ldots, P[2^w - 1]) \leftarrow \texttt{alg}(P, w)$, $\texttt{alg} \in \{\texttt{SmallMultJ}, \texttt{SmallMultA}\}$.
4. $(P[-1], P[-3], \ldots, P[-2^w + 1]) \leftarrow (-P[1], -P[3], \ldots, -P[2^w - 1])$.
5. Represent d as $\sum_{i=0}^{k-1} d_i 2^{wi}$, $d_0, d_1, \ldots, d_{k-1} \in \{0, 1, \ldots, 2^w - 1\}$.
6. $(d_{k-1}, d_{k-2}) \leftarrow (d_{k-1} + \mathrm{even}(d_{k-1}), d_{k-2} - \mathrm{even}(d_{k-1})2^w)$.
7. $Q \leftarrow P[d_{k-1}]$.
8. For $i = k-2, k-3, \ldots, 1$:
    1) $(d_i, d_{i-1}) \leftarrow (d_i + \mathrm{even}(d_i), d_{i-1} - \mathrm{even}(d_i)2^w)$;
    2) $Q \leftarrow 2^w Q$ ($J \leftarrow 2J$, w times);
    3) $Q \leftarrow Q + P[d_i]$ ($J \leftarrow J + J$ or $J \leftarrow J + A$).
9. $Q \leftarrow 2^w Q$.
10. $Q \leftarrow Q + P[d_0]$ ($A \leftarrow J + J$ or $A \leftarrow J + A$).
11. $Q \leftarrow (-1)^\delta Q$.
12. Return Q.

Exceptional cases

Exception #1: addition with O:

- avoided (easy analysis).

Exception #2: doubling instead of addition:

- can be only at Step 10;
- …when $d = q - 2\delta$, $\delta \in \{1, 2, \ldots, 2^w - 1\}$ such that $2^w \mid (q - \delta)$, $2^{w+1} \nmid (q - \delta)$;
- blocked by complete formulas $A \leftarrow J + J/A$ (through the cascade: $H \leftarrow J$, $H \leftarrow H + H/A$, $A \leftarrow H$).

Tuning

Heuristics $h = \left(\frac{\mathtt{I}}{\mathtt{M}}, \frac{\mathtt{I}}{\mathtt{M}}, \frac{\mathtt{A{\approx}m{\approx}h}}{\mathtt{M}}\right)$ **examined**:

- $(100, 0.8, 0)$;
- $(100, 0.67, 0)$;
- $(100, 0.67, 0.05)$.

Bitlen of q: 256, 384, 512.

Analysis (in view of $\mathtt{M}$-complexity):

- Strategy II is better than Strategy I;
- Strategy II can be even faster if we replace $(J \leftarrow 2J, J \leftarrow J + A)$ with $J \leftarrow 2J + A$;
- for Strategy II, the best window length $w^* \in \{4, 5\}$ (8 or 16 small multiples to store).

Comparison

l	Algorithm	M-complexity (rounded to the nearest integer)		
		M(100, 0.8, 0)	M(100, 0.67, 0)	M(100, 0.67, 0.05)
256	ScalarMult[SmallMultJ, 5]	3046	2807	2989
	ScalarMult[SmallMultA, 4]	**2846**	**2636**	**2830**
	MontLadder[WeierCurve]	**2724**	**2590**	**2731**
	MontLadder[MontCurve]	2200	2067	2182
384	ScalarMult[SmallMultJ, 6]	4382	4032	4297
	ScalarMult[SmallMultA, 5]	**4090**	**3774**	**4053**
	MontLadder[WeierCurve]	**4030**	**3829**	**4041**
	MontLadder[MontCurve]	3250	3050	3223
512	ScalarMult[SmallMultJ, 6]	5741	5274	5626
	ScalarMult[SmallMultA, 5]	**5333**	**4912**	**5284**
	MontLadder[WeierCurve]	**5335**	**5068**	**5350**
	MontLadder[MontCurve]	4299	4033	4264

ScalarMult[SmallMultJ, w^*] — Strategy I.

ScalarMult[SmallMultA, w^*] — Strategy II.

MontLadder[MontCurve] — the Montogomery ladder on short Weierstrass curves (M. Hamburg, 2020).

MontLadder[MontCurve] — the Montogomery ladder on Montgomery curves (P. Montgomery, 1987; RFC 7748; the current champion).

Resources

https://github.com/bcrypto/smult