Merging precomputation and scalar multiplication

Nicolas Méloni

February 25th, 2010

Outline

1. Introduction

2. Classic Algorithms

3. The art of recycling

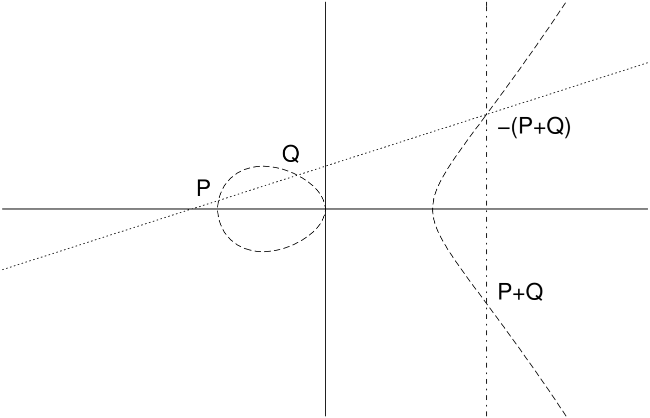
4. Finding better sets of digits

Outline

1. Introduction
2. Classic Algorithms
3. The art of recycling
4. Finding better sets of digits

If you please ... draw me an elliptic curve (over $\mathbb{R}$)

group law of an elliptic curve over $\mathbb{R}$



Elliptic curve defined over a finite field

Definition

- $\mathbb{K}$ a characteristic p field (with $p > 3$)
- $P = (x, y) \in \mathbb{K}^2$ a point satisfaying $y^2 = x^3 + ax + b$, with $a, b \in \mathbb{K}$

Addition formulae

$P_1 = (x_1, y_1), P_2 = (x_2, y_2), P_3 = P_1 + P_2$

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

Point scalar multiplication

Point multiplication

- $k \in \mathbb{N}, P \in E(K)$
- $[k]P = \underbrace{P + \cdots + P}_{k \text{ fois}}$

Main issue

- Performing this operation as fast as possible

Three levels of arithmetic

- Finite field
- Point addition
- Scalar point multiplication

Outline

1 Introduction

2 Classic Algorithms

3 The art of recycling

4 Finding better sets of digits

Fast scalar multiplication

Double-and-add

- $k = \sum_{i=0}^{n-1} k_i 2^i$
- $n-1$ doublings and $n/2$ additions (on average)

Computation of 21P

- $k = 21 = 2^4 + 2^2 + 2^0 = 10101_2 = k_4 k_3 k_2 k_1 k_0$

Double-and-add

- Initialization: $Q \leftarrow P$
- $b_3 = 0 : Q \leftarrow 2Q \qquad (Q = 2P)$
- $b_2 = 1 : Q \leftarrow 2Q + P \qquad (Q = 5P)$
- $b_1 = 0 : Q \leftarrow 2Q \qquad (Q = 10P)$
- $b_0 = 1 : Q \leftarrow 2Q + P \qquad (Q = 21P)$

Non Adjacent Form (NAF)

- $k_i \in \{-1, 0, 1\}$
- $k = 31 = 11111_2 = 10000\overline{1}_{NAF}$
- at most n doublings and $\frac{n}{3}$ additions (on average)

Windows method: wNAF

- $|k_i| < 2^{w-1}$
- $k = 267 = 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1\ 1$
- 2-NAF: $1\ 0\ 0\ 0\ 1\ 0\ \overline{1}\ 0\ \overline{1}$
- 3-NAF: $1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 3$
- 4-NAF: $1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ \overline{5}$
- 5-NAF: $1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 11$

Method

- Compute and store $P, 3P, \ldots, 2^{w-1}P$
- Perform a Double-and-Add like scheme
- # of additions: $n/(w+1)$
- # of doublings: $n-w+1$

Going from wNAF to $(w+1)$NAF

- Twice as much precomputed points
- # of additions decreased by a factor $w/(w+1)$
- # of doublings decreased by 1!

Outline

1. Introduction

2. Classic Algorithms

3. The art of recycling

4. Finding better sets of digits

Simple example: $k = 10011k_{n-5}\dots k_0$

Using dictionary $\{1, 3, 5\}$

- recoding $k = 1003k_{n-5}\dots k_0$
- compute $2P, 3P, 5P$
- $(100)_2P = 8P, (1000)_2P = 16P, (1003)_2P = 19P$
- finish with double-and-add

Recycling approach

- compute $2P, 4P, 8P, 9P, 18P$ and $19P$ using double-and-add
- store $P, 9P, 19P$
- use dicionary $\{1, 9, 19\}$ during the recoding of k
- finish with double-and-add

Comparisons

Using dictionary $\{1, 3, 5\}$

- three point doublings and three point additions
- average density 2/9

Using dictionary $\{1, 9, 19\}$

- four point doublings and two point additions
- average density 2/9

We have traded a point addition for a doubling!

Outline

1. Introduction
2. Classic Algorithms
3. The art of recycling
4. Finding better sets of digits

Finding better sets of digits

Using Double-and-Add

- $\{1, 2, 4, 8, 9, 18, 19\}$ does not provide a *good* set because of too many doublings
- $2{=}(10)_2$, $4{=}(100)_2$ and $8{=}(1000)_2$ are not useful

Idea

- Use chains with more additions
- Choice: euclidean addition chains
- $\{1, 2, 3, 5, 7, 12, 19\}$ lead to the dictionary $\{1, 3, 5, 7, 19\}$
- average density $4/21(\simeq 0.19) < 2/9(\simeq 0.22)$

New Scalar Multiplication Scheme

$k = (k_{t-1} \dots k_0)_2$

- $k_H = \lfloor k/2^{t-l} \rfloor$ for some l
- $k_L = k - k_H * 2^{t-l}$
- find short Euclidean chain $S = (s_1, \dots s_m)$ computing k_H
- use S as a dictionary for the recoding of
- compute $n_H P$ using chain S and store the values $P, s_1 P, s_2 P, \dots, s_m P$
- compute $(k_H * 2^{t-l} + k_L)P$ using the double-and-add alg. starting from $k_H P$.

Example

$k = 1626832774 = 193 \times 2^{23} + 7831430$

- $k_H = 193$
- $k_L = 7831430$
- $S = 1, 2, 3, 5, 7, 12, 19, 31, 50, 81, 112, 193$
- $k_L = (5\ 0\ 0\ 0\ 0\ 0\ \text{-}81\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ \text{-}1\ 0\ 0\ 0\ 0\ 0\ 3\ 0)_2$
- compute $193P$ using chain S and store $P, 3P, 5P$ and$81P$
- compute $193 \times 2^{23}P + 7831430P$ using the double-and-add alg. starting from $193P$.

The end

Any questions?