Elliptic Curve Timing

Maria Polyakova

June 16, 2013

Background

For the assignment, the speed of operations on elliptic curves in different coordinates will be explored. An elliptic curve is defined to be $y^2 = x^3 + ax + bmodp$, and repress a group, over which certain operations can be performed. The operations can take place in several coordinates, which may simplify and quicken some operations.
Two coordinate systems that were trialed here are the affine coordinates and the projective coordinates.

Affine Coordinates

Addition in affine coordinates is defined as follows:
Given two points (x_1, y_1) and (x_2, y_2) on an elliptic curve, addition output a third point (x_3, y_3) with the following properties:

$$x_3 = s - x_1 - x_2 mod(p)$$

$$y_3 = s(x_1 - x_3) - y_1$$

if the two points being added are the same,

$$s = \frac{3x_1^2 + a}{2y_1} mod(p)$$

when points are not the same

$$s = \frac{y_2 - y_1}{x_2 - x_1} mod(p)$$

This is a fairly long operation, because it requires taking inverses. Taking an inverse of a number mod(p) is $\lambda^{-1} mod(p) = \lambda^{p-2} mod(p)$, which takes a long time. That is why frequently, it is easier to use projective coordinates.

Projective Coordinates

Points in projective coordinates have 3 values, not two. The point (X_1, Y_1, Z_1) corresponds to the affine point $(\frac{X_1}{Z_1^2}, \frac{Y_1}{Z_1^3})$, when Z_1 is unequal to zero. Otherwise, it is the point at infinity. When converting to projective, Z is simply given the value of 1 at the start. Addition is defined as:

$$A = X_1 Z_2^2 \; B = X_2 Z_1^2 \; C = Y_1 Z_2^3 \; D = Y_2 Z_1^3 \; E = B - A \; F = D - C$$

and

$$X_3 = -E^3 - 2AE^2 + F^2 \; Y_3 = -CE^3 + F(AE^2 - X^3) \; Z_3 = Z_1 Z_2 E$$

Doubling(when the points are the same) is defined as:

$$A = 4X_1 Y_1^2 \; B = 3X_1 + aZ_1^4$$

and

$$X_3 = -2A + B^2 \; Y_3 = -8Y_1^4 + B(A - X_3) \; Z_3 = 2Y_1 Z_1$$

Because the following operations don't require an inversion, they are much faster than the affine coordinates.

Curves

Because complexity is defined by how large each curve is, for cryptographic purposes, the curves must be very large. The curve is defined as above, and the parameters are given by NIST, and claimed as secure. The curve used is below.
The Curve is P-192
$p - 192 = 6277101735386680763835789423207666416083908700390324961279$
$a = 6277101735386680763835789423207666416083908700390324961276$
$b = 245515554600894381774029391519745178476910805816119123806 5$

Generating Point P:
$x_P = 602046282375688656758213480587526111916698976636884684818$
$y_P = 174050332293622031404857552280219410364023488927386650641$
order $n = 6277101735386680763835789423176059013767194773182842284081$

Timings: Implemented in Python

For each of the values of the base, the affine timing should be slower than the projective. The larger the base, the longer exponentiation will take, and therefore, the greater the difference that should be seen between operations done in affine coordinates and the operations done in projective coordinates. For any value, the projective coordinates should be much faster than affine. Below is a timing of the multiplication on elliptic curves in projective and affine coordinates. d and b are the constants by which each point must be multiplied. Base values, 2^{bit} -1

192-bit
$d_1 = 627710173538668076383578942320766641610235544446403451289 5$

144-bit
$d_2 = 223007451985306231415357182726483615059804 15$

96-bit
$d_3 = 63382530011411470074835160 2687$

Edited Values, to mess up the signed-digit expansion
192-bit
$b_1 = 627710173538668086383528942329766641610235544446403451289 5$

144- bit
$b2 = 223007451988396231415357182726483215059804 15$

96-bit
$b3 = 69382530011411480074835163 2687$

$[d]$	Affine(sec)	Projective(sec)
d_1	.174	.073
b_1	.142	.092
d_2	.119	.086
b_2	.121	.073
d_3	.101	.069
b_3	.109	.063

As expected, multiplication in affine coordinates is consistently smaller than in projective coordinates.