Elliptic Curve Cryptography

- majority of public-key crypto (RSA, D-H) use either integer or polynomial arithmetic with very large numbers/polynomials
- imposes a significant load in storing and processing keys and messages
- an alternative is to use elliptic curves
- offers same security with smaller bit sizes

Abelian Group

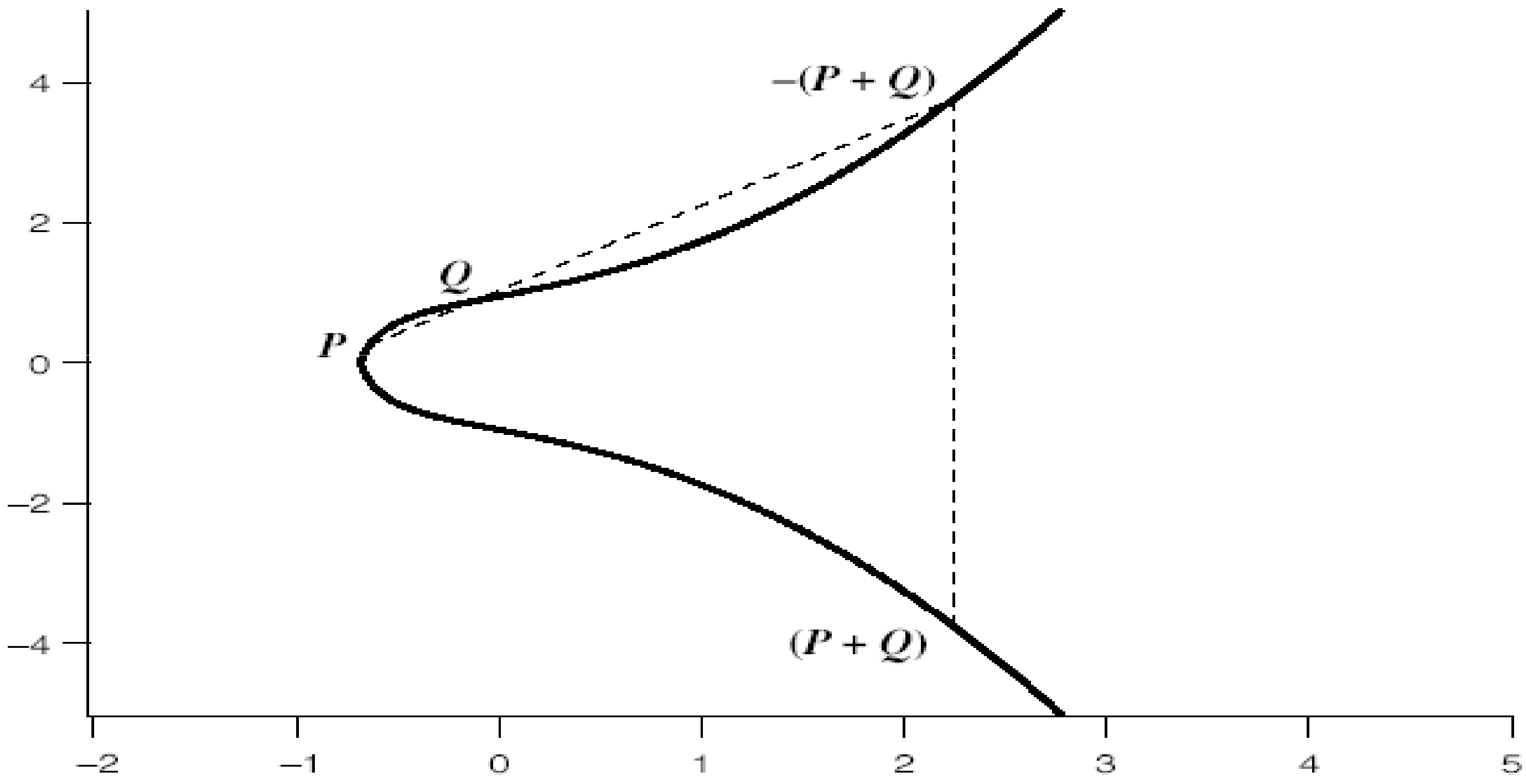
{G, **.**} – a set of elements with binary operation **.**

- Closure
- Associative
- Identity element
- Inverse element
- Commutative

Elliptic Curves over Real Numbers

- an elliptic curve is defined by an equation in two variables x & y, with coefficients
- consider a cubic elliptic curve of form
    - $y^2 = x^3 + ax + b$
        - where x,y,a,b are all real numbers
    - also define zero point or point at infinity, O
- To plot the curve, we compute
    - $y = \sqrt{x^3 + ax + b}$
- For given a and b, the plot consists of positive and negative values of y for each value of x
- have addition operation for elliptic curve
    - geometrically sum of P+Q is reflection of intersection R

Real Elliptic Curve Example



(b) $y^2 = x^3 + x + 1$

Geometric Description of Addition

- A group can be defined based on the set E(a,b) provided that $\sqrt{x^3 + ax + b}$ has no repeated factors.

  i.e $4a^3+27b^2 \neq 0$

- If 3 points lie on a straight line, their sum is 0.

Rules of addition

1. P + O = P where P is (x,y)
2. P + (-P) = O where –P is (x,-y)
3. P + Q be the mirror image of the third point of intersection where P and Q are two points with different x coordinates
4. If P and –P are points with same x coordinate , they can be joined by a vertical line, thus P + (-P) = O
5. Doubling a point Q, i.e Q + Q = 2Q = -S

Algebraic Description of Addition

- $P=(x_p,y_p)$ ad $Q=(x_q,y_q)$
- Slope of line l that joins them is

  $\Delta = (y_q - y_p) / (x_q - x_p)$
- $R = P + Q$

  $x_R = \Delta^2 - x_p - x_q$

  $y_R = - y_p + \Delta(x_p - x_R)$
- $P + P = 2P = R$ When $y_p \neq 0$

  $x_R = (3x^2_p + a / 2y_p)^2 - 2 x_p$

  $y_R = (3x^2_p + a / 2y_p) (x_p - x_R) - y_p$

Finite Elliptic Curves

- Elliptic curve cryptography uses curves whose variables & coefficients are finite
- have two families commonly used:
  - prime curves $E_p(a,b)$ defined over Z_p
    - use integers modulo a prime
    - best in software
  - binary curves $E_{2m}(a,b)$ defined over $GF(2^n)$
    - use polynomials with binary coefficients
    - best in hardware

Elliptic Curve Cryptography

- ECC addition is analog of modulo multiply
- ECC repeated addition is analog of modulo exponentiation
- need “hard” problem equiv to discrete log
  - `Q=kP`, where Q,P belong to a prime curve
  - is “easy” to compute Q given k,P
  - but “hard” to find k given Q,P
  - known as the elliptic curve logarithm problem
- Certicom example: $E_{23}(9,17)$

ECC Key Exchange

- can do key exchange analogous to D-H
- users select a suitable curve $E_p(a,b)$
- select base point $G=(x_1,y_1)$ with large order n such that $nG=O$
- A & B select private keys $n_A<n$, $n_B<n$
- compute public keys: $P_A=n_A\times G$, $P_B=n_B\times G$
- compute shared key: $K=n_A\times P_B$, $K=n_B\times P_A$
  - same since $K=n_A\times n_B\times G$

ECC Encryption/Decryption

- several alternatives, will consider simplest
- must first encode any message M as a point on the elliptic curve P_m
- select suitable curve & point G as in D-H
- each user chooses private key $n_A<n$
- and computes public key $P_A=n_A \times G$
- to encrypt P_m : $C_m=\{kG,\ P_m+k\ P_B\}$, k random
- decrypt C_m compute:

$$P_m+kP_B-n_B(kG) = P_m+k(n_BG)-n_B(kG) = P_m$$

ECC Security

- relies on elliptic curve logarithm problem
- fastest method is “Pollard rho method”
- compared to factoring, can use much smaller key sizes than with RSA etc
- for equivalent key lengths computations are roughly equivalent
- hence for similar security ECC offers significant computational advantages