

Modular Inverse:

$$\text{GCD}(a, b) = \text{GCD}(b, a \bmod b). \quad 1024$$

Time $O(\log b)$. Euclidean algorithm for finding gcd.

$$b = a^{-1} \bmod N$$
$$\Rightarrow ab = 1 \bmod N.$$

$$\text{gcd}(a, N) = 1.$$

$$ax + Ny = 1$$
$$ax \bmod N$$

$$b_{i+2} \leq b_i / 2$$

$$(a_0, b_0) \rightarrow (a_1, b_1) \dots$$

$$b = 1111$$
$$15$$

Extended Euclidean algorithm:-

Input: Integers a, b with $a \geq b > 0$.

Output: (d, x, y) with $d = \gcd(a, b)$ and $ax + by = d$.

if $b \mid a$

return $(b, 0, 1)$.

$e\text{GCD}(a, b)$

$e\text{GCD}(b, a \bmod b)$

$$a = bq + (a \bmod b)$$

else

$(d', x', y') \leftarrow e\text{GCD}(b, a \bmod b) \Rightarrow \gcd(b, a \bmod b) = bx' + (a \bmod b)y'$

return $(d, y', x' - y' \lfloor a/b \rfloor)$

$$d' = bx' + (a \bmod b)y'$$

$$= bx' + (a - b \lfloor a/b \rfloor)y'$$

$$= ay' + b(x' - \lfloor a/b \rfloor y')$$

$$\text{egcd}(18, 12).$$

$$\leftarrow \text{egcd}(12, 6).$$

$$(d, x, y).$$

$$(6, 0, 1)$$

$$6 = 12x + 6y$$

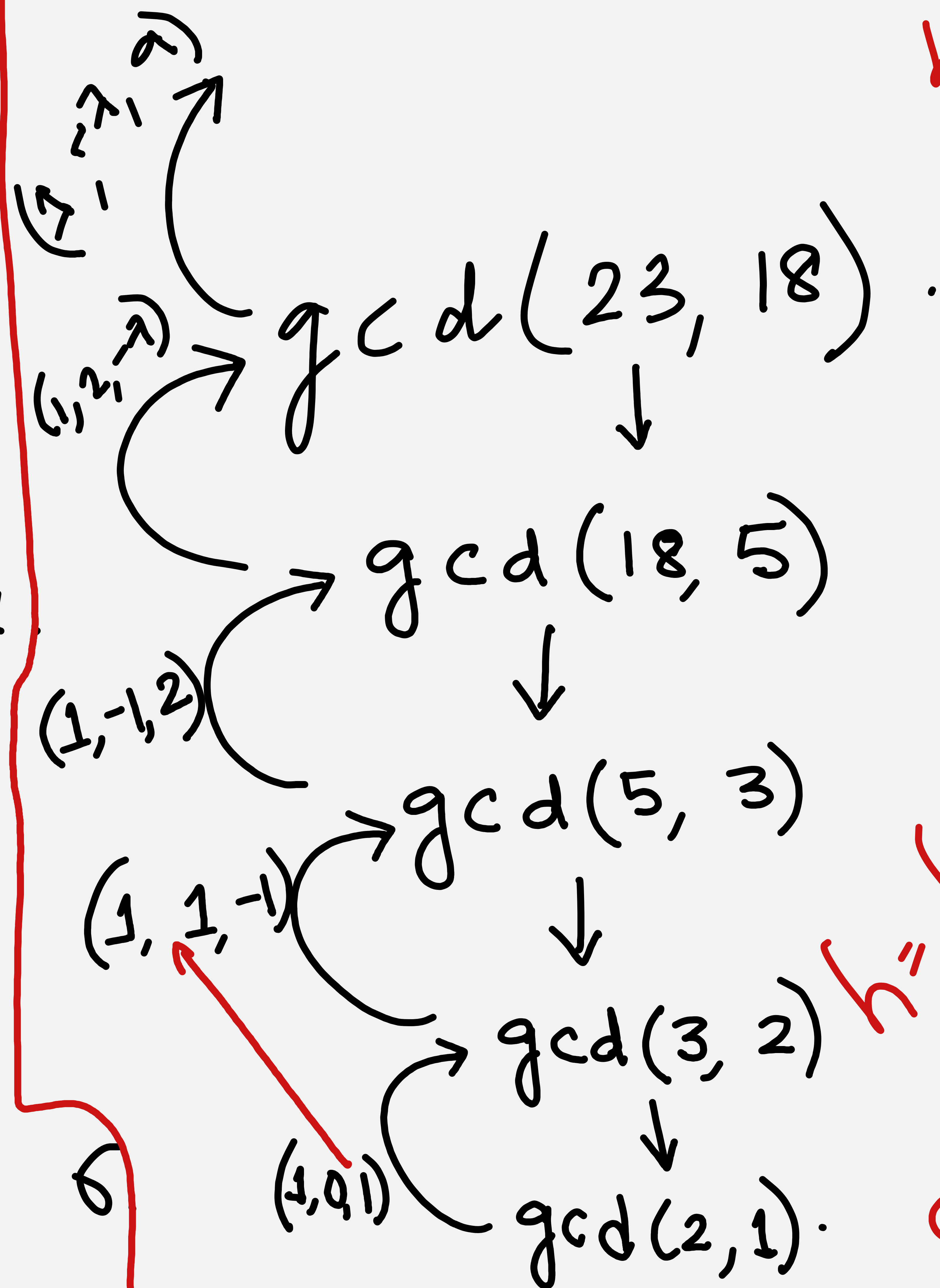
$$\Rightarrow 6 = 12 \cdot 0 + 6 \cdot 1$$

$$x = 0, y = 1.$$

$$6 = 18x + 12y.$$

$$18 \cdot 1 + 12(-1) = 6$$

$$(x=1, y=-1)$$



$$b_0 - b_1 - \textcircled{b_2} - b_3 - \textcircled{b_4} - b_5 = \textcircled{b_6}$$

$$= b_0/2 \quad b_0/2^2 \quad b_0/2^3$$

$$\textcircled{2^{+n}}$$

$$2^{-(n)} \cdot 1 = a \cdot \textcircled{b_i}$$

$$b_0/2^i = b_i$$

$$y', x' - y' \textcircled{b_i}$$

$$1 - (-1) \cdot 1$$

$$= 1 + 1 = 2.$$

$$-1 - 2 \times 3$$

$$= -1 - 6 = -7.$$

$$0 - 1 \cdot 1$$

$$= -1$$

Modular Exponentiation:-

We are given a base $a \in \mathbb{Z}_N$.
and an exponent $b > 0$.

$$a^b \bmod N = a \cdot a^{b-1} \bmod N$$

Compute $a^b \bmod N$.

$$O(b) = O\left(2^{\log_2 b}\right)$$

$$N = 10, a = 3, b = 7$$

$$3 \bmod 10, 3^2 \bmod 10, 3^3 \bmod 10, \dots, 3^7 \bmod 10$$

Base: a

Exponent = $b > 0$

Modulus = N

$a^b \bmod N$: Previous Recurrence.

$$a^b \bmod N = a \cdot a^{b-1} \bmod N.$$

Suppose b is even

$$a^b \bmod N = \begin{cases} (a^{b/2})^2 \bmod N & \text{if } b \text{ is even} \\ a \cdot (a^{b/2})^2 \bmod N & \text{if } b \text{ is odd} \end{cases}$$

Repeated Squaring / Square and Multiplication

Input: Modulus N , base $a \in \mathbb{Z}_N$, exponent $b > 0$

Output: $a^b \bmod N$.

$x := a$.

$t := 1$.

While ($b > 0$).

if b is odd.

$\begin{cases} t := t \cdot x \bmod N \\ b := b - 1 \end{cases}$

end if

$x := x^2 \bmod N$

$b := b/2$.

return t .

$a = 3, b = 7, N = 10$

$k = 7$.

1st it: $t = 3, b = 6, x = 9, b = 3$.

2nd it: $t = 7, b = 2, x = 1, b = 1$

3rd it: $t = 7, b = 0, x = 1, b = 0$.

If we know base a , and a priori bound

on the length of b , say $\|b\| \leq n$, then

using some preprocessing, one can improve

the run time of the algorithm.

$$f_{a,N}(b) = a^b \bmod N$$

$$a^b = a^{\sum_{i=0}^{n-1} b_i 2^i}$$

$$= (a^{2^0})^{b_0} \cdot (a^{2^1})^{b_1} \cdots (a^{2^{n-1}})^{b_{n-1}}$$

$$x_0 = a^{2^0}, x_1 = a^{2^1}, x_2 = a^{2^2} \cdots x_{n-1} = a^{2^{n-1}}$$

$$a^b = \prod_{i=0}^{n-1} x_i^{b_i}$$

$$b_i \in \{0, 1\}.$$

Remark:- $f_{a,N}(b) = a^b \bmod N$ is easy to compute.

Computing the inverse is difficult, known as discrete log problem.

$$\frac{|G|}{2^n}$$

Failure prob in a trial: $\left(1 - \frac{|G|}{2^n}\right)$

$$\left(1 - \frac{|G|}{2^n}\right)^t$$

$$1 - \left(1 - \frac{|G|}{2^n}\right)^t$$

$$2^{n-1} < |G| < 2^n$$

$$\left(1 - \frac{|G|}{2^n}\right)^t \approx \left(\frac{1}{2}\right)^t$$

$$P[A \wedge B] = P[A] \cdot P[B]$$

$$P[A|B] = P[A]$$

$$= \frac{P[A \wedge B]}{P[B]} = P[A]$$

$$P[E_1^c \wedge E_2^c \wedge E_3^c \wedge \dots \wedge E_t^c]$$

$$= \prod_{i=1}^t P[E_i^c] = \left(1 - \frac{|G|}{2^n}\right)^t \leq \frac{1}{2^t}$$

$$t = O(n)$$

$$= \frac{1}{2^n}$$

