Modular arithmetic

AiC FME, UPC

Fall 2023

Review of Modular Arithmetic

Given $a, b, n \in \mathbb{Z}$, a is congruent to b modulo n ($a \equiv b \mod n$) if $n|(a-b)$.

Review of Modular Arithmetic

Given $a, b, n \in \mathbb{Z}$, a is congruent to b modulo n ($a \equiv b \mod n$) if $n|(a-b)$.

- If $a \mod n = b$ then $a \equiv b \mod n$.
- $(a+b) \mod n \equiv ((a \mod n) + (b \mod n)) \mod n$.
- $(a \cdot b) \mod n \equiv ((a \mod n) \cdot (b \mod n)) \mod n$.

Review of Modular Arithmetic

Given $a, b, n \in \mathbb{Z}$, a is congruent to b modulo n ($a \equiv b \mod n$) if $n|(a - b)$.

- If $a \mod n = b$ then $a \equiv b \mod n$.
- $(a + b) \mod n \equiv ((a \mod n) + (b \mod n)) \mod n$.
- $(a \cdot b) \mod n \equiv ((a \mod n) \cdot (b \mod n)) \mod n$.

n partitions $\mathbb{Z}$ in n equivalence classes:

$$\mathbb{Z}_n = \{0, 1 \ldots, n - 1\}.$$

Review of Modular Arithmetic

Given $a, b, n \in \mathbb{Z}$, a is congruent to b modulo n ($a \equiv b \mod n$) if $n|(a-b)$.

- If $a \mod n = b$ then $a \equiv b \mod n$.
- $(a+b) \mod n \equiv ((a \mod n) + (b \mod n)) \mod n$.
- $(a \cdot b) \mod n \equiv ((a \mod n) \cdot (b \mod n)) \mod n$.

n partitions $\mathbb{Z}$ in n equivalence classes:

$$\mathbb{Z}_n = \{0, 1 \ldots, n-1\}.$$

On $\mathbb{Z}_n$, we define operators $+_n, \cdot_n$ as $+, \cdot \mod n$.

Review of Modular Arithmetic

Recall that $(\mathbb{Z}_n, +_n,)$ form a abelian group.

Review of Modular Arithmetic

Recall that $(\mathbb{Z}_n, +_n,)$ form a abelian group.

In fact $(\mathbb{Z}_n, +_n, \cdot_n)$ form a commutative ring, therefore:

1. $a + (b + c) \equiv (a + b) + c \mod n$ (associativity)
2. $ab \equiv ba \mod n$ (commutativity)
3. $a(b + c) \equiv ab + ac \mod n$ (distributivity)

Review of Modular Arithmetic

Recall that $(\mathbb{Z}_n, +_n,)$ form a abelian group.

In fact $(\mathbb{Z}_n, +_n, \cdot_n)$ form a commutative ring, therefore:

1. $a + (b + c) \equiv (a + b) + c \mod n$ (associativity)
2. $ab \equiv ba \mod n$ (commutativity)
3. $a(b + c) \equiv ab + ac \mod n$ (distributivity)

These operations can help in simplifying big calculations.

For example to compute $2^{285} \mod 31$:

$$2^{285} \equiv (2^5)^{57} \equiv 32^{57} \equiv 1^{57} \equiv 1 \mod 31$$

Modular multiplication

Modular multiplication: Given $x, y, n \in \mathbb{N}$, compute $x\ y \mod n$.

Modular multiplication

Modular multiplication: Given $x, y, n \in \mathbb{N}$, compute $x\ y \mod n$.

- We first perform the product $x\ y$ and then take modulo n which take $O(M^2)$ where $M = \max\{\lg x, \lg y, \log n\}$.

Modular multiplication

Modular multiplication: Given $x, y, n \in \mathbb{N}$, compute $x\ y \mod n$.

- We first perform the product $x\ y$ and then take modulo n which take $O(M^2)$ where $M = \max\{\lg x, \lg y, \log n\}$.
- If $x, y \leq n$, and $N = \lg N$, the cost is $O(N^2)$.

Modular exponentiaton

Modular exponentiation: given $a, b, n \in \mathbb{N}$, compute $d = a^b \mod n$.

Modular exponentiaton

Modular exponentiation: given $a, b, n \in \mathbb{N}$, compute $d = a^b \mod n$.

- Repeating squaring according to binary representation of exponent (D&C).

Modular exponentiaton

Modular exponentiation: given $a, b, n \in \mathbb{N}$, compute $d = a^b \mod n$.

- Repeating squaring according to binary representation of exponent (D&C).
- The algorithm is tuned to reduce modulo n at each operation, in particular it relies in 2 facts:
  1. $a \cdot b \mod n = (a \mod n) \cdot (b \mod n)$,
  2. $a^{2c} = (a^c)^2$.

Modular exponentiaton

Modular exponentiation: given $a, b, n \in \mathbb{N}$, compute $d = a^b \mod n$.

- Repeating squaring according to binary representation of exponent (D&C).
- The algorithm is tuned to reduce modulo n at each operation, in particular it relies in 2 facts:
  1. $a \cdot b \mod n = (a \mod n) \cdot (b \mod n)$,
  2. $a^{2c} = (a^c)^2$.
- For example, to compute a^{1101}:
  $a^1 \to a^{10} \to a^{11} \to a^{110} \to a^{1100} \to a^{1101}$.
  i.e. $a \to a^2 \to a^3 \to a^6 \to a^{12} \to a^{13}$.

Algorithm for $d = a^b \mod N$.

Let $N = \lg b$ and assume $N \geq \max\{\lg a, \lg n\}$,

Expo$(a, b[N-1, \dots, 0], n)$
$d = 1$
for $i = N - 1$ down to 0
do
  $d = (d \cdot d) \mod n$
  if $b[i] == 1$ **then**
    $d = (d \cdot a) \mod n$
  end if
end for
return d

Algorithm for $d = a^b \mod N$.

Let $N = \lg b$ and assume $N \geq \max\{\lg a, \lg n\}$,

Expo$(a, b[N-1,\dots,0], n)$
$d = 1$
for $i = N-1$ down to 0
do
  $d = (d \cdot d) \mod n$
  if $b[i] == 1$ **then**
    $d = (d \cdot a) \mod n$
  end if
end for
return d

Complexity: The number of loops is $N = \lg b$. At each loop, the algorithm does a constant number of multiplications and may be a shifting.

The bit complexity is $O(N(\lg n)^2) = O(N^3)$.

Example

Wish to compute $3^{13} \mod 5$

$b = 13, a = 3, N = 5,$

$b = 13 \Rightarrow b = (1101)_2 \Rightarrow 13 = 2^3 + 2^2 + 2^0$

So $3^{13} \mod 5 = 3^{2^3+2^2+2^0} \mod 5$
$= (((3^2)^3 \mod 5) \cdot ((3^2)^2 \mod 5) \cdot ((3^2)^0 \mod 5) \mod 5)$

i	$b[i]$		d
3	1	$1 \mod 5 = 1$; $3 \mod 5 = 3$	3
2	1	$9 \mod 5 = 4$; $12 \mod 5 = 2$	2
1	0	$4 \mod 5 = 4$	4
0	1	$16 \mod 5 = 1$; $3 \mod 5 = 3$	3

The Discrete LOG

Consider the following *backwards* version of modular exponentiation:

Discrete Log: given $a, b, n \in \mathbb{Z}^+$, exists $y \in \mathbb{Z}^+$ such that $a = b^y \mod n$?

The Discrete LOG

Consider the following *backwards* version of modular exponentiation:

Discrete Log: given $a, b, n \in \mathbb{Z}^+$, exists $y \in \mathbb{Z}^+$ such that $a = b^y \mod n$?

- Difficult problem! it is not known to be in P neither NP-hard.
- Given $a, b, n \in \mathbb{Z}^+$ to compute $x = a^b \mod n$ can be done in $O(N^3)$ steps ($N = |b|$).
- Given $a, b, n \in \mathbb{Z}^+$ to compute y s.t. $a = b^y \mod n$ id difficult.

Modular inverse

Modular inverse: Given $x, n \in \mathbb{N}$, compute y such that $x/y \equiv 1 \mod n$.

- $x/y \mod n$ does not always exists.
- For ex. In Z_{15}, 3 does not have inverse in Z_{15}. For $a \in Z_n$, a^{-1} exists in Z_n if $\gcd(a, n) = 1$.

Modular inverse

Modular inverse: Given $x, n \in \mathbb{N}$, compute y such that $x/y \equiv 1 \mod n$.

- $x/y \mod n$ does not always exists.
- For ex. In Z_{15}, 3 does not have inverse in Z_{15}. For $a \in Z_n$, a^{-1} exists in Z_n if $\gcd(a, n) = 1$.

Define, $\mathbb{Z}_n^* = \{a | a \in \{1, 2, \dots, n-1\} \wedge \gcd(a, n) = 1\}$.

- $\mathbb{Z}_n^*$ is the set of relative primes with n.
- Example: $\mathbb{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$
- $(\mathbb{Z}_n^*, \cdot_n)$ is an abelian group.

Modular inverse

Modular inverse: Given $x, n \in \mathbb{N}$, compute y such that $x/y \equiv 1 \mod n$.

- $x/y \mod n$ does not always exists.
- For ex. In Z_{15}, 3 does not have inverse in Z_{15}. For $a \in Z_n$, a^{-1} exists in Z_n if $\gcd(a, n) = 1$.

Define, $\mathbb{Z}_n^* = \{a | a \in \{1, 2, \dots, n-1\} \wedge \gcd(a, n) = 1\}$.

- $\mathbb{Z}_n^*$ is the set of relative primes with n.
- Example: $\mathbb{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$
- $(\mathbb{Z}_n^*, \cdot_n)$ is an abelian group.

To solve the problem we need to check if the inverse exists and then compute the inverse efficiently.

Greatest common divisor

GCD: given $a, b \in \mathbb{Z}^+$, compute $\gcd(a, b)$

Recall that given $a, b \in\in \mathbb{Z}^+$, the $\gcd(a, b)$ is the largest integer which divides a and b.

We can compute gcd using Euclid's algorithm.

Euclid's algorithm for GCD.

Theorem (Euclid)

For any $a, b \in \mathbb{Z}$ *with* $b > 0$, $\gcd(a, b) = \gcd(b, a \mod b)$.

EUCLID(a, b)
if $b = 0$ **then**
  return a
else if $b = 1$ **then**
  return 1
else
  EUCLID$(b, a \mod b)$
end if

Each recursive call reduces a at least for $1/2$.
Now if a and b have N bits, the division takes $O(N^2)$ steps. So, the cost is $O(N^3)$.

EUCLID(30,21)→EUCLID(21,9)→ EUCLID(9,3)→ EUCLID(3,0)= 3

Extended Euclid algorithm

Recall Bezout Identity:

Given $a, b \in \mathbb{Z}$, $\exists x, y \in \mathbb{Z}$ s.t. $\gcd(a, b) = d = ax + by$.

Extended Euclid algorithm

Recall Bezout Identity:
Given $a, b \in \mathbb{Z}$, $\exists x, y \in \mathbb{Z}$ s.t. $\gcd(a, b) = d = ax + by$.

The extended algorithm takes as input $a, b \in \mathbb{Z}$ and returns $d \in \mathbb{Z}$ s.t. $d = \gcd(a, b)$, and $x, y \in \mathbb{Z}$ s.t. $d = ax + by$.

Extended Euclid algorithm

Recall Bezout Identity:
Given $a, b \in \mathbb{Z}$, $\exists x, y \in \mathbb{Z}$ s.t. $\gcd(a, b) = d = ax + by$.

The extended algorithm takes as input $a, b \in \mathbb{Z}$ and returns $d \in \mathbb{Z}$ s.t. $d = \gcd(a, b)$, and $x, y \in \mathbb{Z}$ s.t. $d = ax + by$.

EXT-EUCLID(a, b)
if $b = 0$ **then**
  return $(a, 1, 0)$
else
  $(d, x', y') :=$ **EXT-EUCLID** $(b, a \mod b)$
  return $(d, y', x' - \lfloor a/b \rfloor y')$
end if

EXT-EUCLID: example

Example: EXT-EUCLID(99,78)
$(d, x_1, y_1) :=$ **EXT-EUCLID** $(99, 78) = (3, -11, 14)$
$(d, x_2, y_2) :=$ **EXT-EUCLID** $(78, 21) = (3, 3, -11)$
$(d, x_3, y_3) :=$ **EXT-EUCLID** $(21, 15) = (3, -2, 3)$
$(d, x_4, y_4) :=$ **EXT-EUCLID** $(15, 6) = (3, 1, -2)$
$(d, x_5, y_5) :=$ **EXT-EUCLID** $(6, 3) = (3, 0, 1)$
$(d, x_6, y_6) :=$ **EXT-EUCLID** $(3, 0) = (3, 1, 0)$

Therefore $\gcd(99, 78) = 3 = 99 \cdot 11 - 78 \cdot 14$.

EXT-EUCLID: correctness

Theorem

EXT-EUCLID(a, b) *returns* (d, x, y) *s.t.* $\gcd(a, b) = d = ax + by$, *in* $O(N^3)$ *operations.*

EXT-EUCLID: correctness

Theorem

EXT-EUCLID(a, b) returns (d, x, y) s.t. $\gcd(a, b) = d = ax + by$, in $O(N^3)$ operations.

- If $b = 0$, $d = a$ and $a = a \cdot 1 + b \cdot 0$.

EXT-EUCLID: correctness

Theorem

EXT-EUCLID(a, b) *returns* (d, x, y) *s.t.* $\gcd(a, b) = d = ax + by$*, in* $O(N^3)$ *operations.*

- If $b = 0$, $d = a$ and $a = a \cdot 1 + b \cdot 0$.
- Assume that $d = \gcd(a, b) = \gcd(b, a \mod b)$ and that $d = b \cdot x' + (a \mod b) \cdot y'$

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

EXT-EUCLID: correctness

Theorem

EXT-EUCLID(a, b) *returns* (d, x, y) *s.t.* $\gcd(a, b) = d = ax + by$, *in* $O(N^3)$ *operations.*

- If $b = 0$, $d = a$ and $a = a \cdot 1 + b \cdot 0$.
- Assume that $d = \gcd(a, b) = \gcd(b, a \mod b)$ and that $d = b \cdot x' + (a \mod b) \cdot y'$

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

- The cost of the algorithm is the same as EUCLID.

Modular linear equations: $ax \equiv b \mod n$.

Recall:

- $ax \equiv b \mod n$ is solvable for x iff $\gcd(a, n) | b$.
- $ax \equiv b \mod n$ either has $\gcd(a, n)$ distinct solutions mod n or it has no solutions.

Modular linear equations: $ax \equiv b \mod n$.

Recall:

- $ax \equiv b \mod n$ is solvable for x iff $\gcd(a, n)|b$.
- $ax \equiv b \mod n$ either has $\gcd(a, n)$ distinct solutions mod n or it has no solutions.

To find, if any, a solution to if $ax \equiv b \mod n$, we use the following

- Let $d = \gcd(a, n)$, use **EXT-EUCLID** to find $d = ax' + by'$.

Modular linear equations: $ax \equiv b \mod n$.

Recall:

- $ax \equiv b \mod n$ is solvable for x iff $\gcd(a, n) | b$.
- $ax \equiv b \mod n$ either has $\gcd(a, n)$ distinct solutions mod n or it has no solutions.

To find, if any, a solution to if $ax \equiv b \mod n$, we use the following

- Let $d = \gcd(a, n)$, use **EXT-EUCLID** to find $d = ax' + by'$.
  If $d | b$ then $ax \equiv b \mod n$ has as one solution

$$x_0 = x'(b/d) \mod n.$$

Modular linear equations: $ax \equiv b \mod n$.

Recall:

- $ax \equiv b \mod n$ is solvable for x iff $\gcd(a, n) | b$.
- $ax \equiv b \mod n$ either has $\gcd(a, n)$ distinct solutions mod n or it has no solutions.

To find, if any, a solution to if $ax \equiv b \mod n$, we use the following

- Let $d = \gcd(a, n)$, use **EXT-EUCLID** to find $d = ax' + by'$.
  If $d | b$ then $ax \equiv b \mod n$ has as one solution

$$x_0 = x'(b/d) \mod n.$$

- If $ax \equiv b \mod n$ is solvable and has x_0 as first solution, then the equation has d distinct solutions (mod n) given by

$$x_i = x_0 + i(n/d) \text{ for } 0 \leq i \leq d - 1$$

Solution of $ax \equiv b \mod n$

The next algorithm takes as input $a, b, n \in \mathbb{Z}$ and returns all solutions of the equation $ax \equiv b \mod n$.

SOLVE (a, b, n)
$(d, x', y') =$ **EXT-EUCLID** (a, n)
if $d|b$ **then**
  $x_0 = x'(b/d) \mod n$
  for $i = 1$ to $d - 1$ **do**
    return $(x_0 + i(n/d)) \mod n$
  end for
else
  return no solution
end if

Complexity: If $N = \max\{\log n, \log a, \log b\}$, then $T(n) = O(N^3)$.

Example

Solve $14x \equiv 30 \mod 100$

SOLVE$(14, 30, 100)$

as $d = 2|30$:

- $(d, x', y') = (2, -7, 1)$
- $x_0 = (-7)(15) \mod 100 = 95$
- $x_1 = 95 + 50 \mod 100 = 45$
- *Solutions: 45, 95*

Finding the multiplicative inverse of $a \in \mathbb{Z}_n^*$

Modular division: given $x, y, n \in \mathbb{N}$, compute $x/y \mod n$ (if it exists!).

Finding the multiplicative inverse of $a \in \mathbb{Z}_n^*$

Modular division: given $x, y, n \in \mathbb{N}$, compute $x/y \mod n$ (if it exists!).

- As $(\mathbb{Z}_n^*, \cdot)$ is an abelian group, then $\forall a \in \mathbb{Z}_n^*, \exists a^{-1} \in \mathbb{Z}_n^*$ the multiplicative inverse such that $a \cdot a^{-1} \equiv 1 \mod n$

Finding the multiplicative inverse of $a \in \mathbb{Z}_n^*$

Modular division: given $x, y, n \in \mathbb{N}$, compute $x/y \mod n$ (if it exists!).

- As $(\mathbb{Z}_n^*, \cdot)$ is an abelian group, then $\forall a \in \mathbb{Z}_n^*, \exists a^{-1} \in \mathbb{Z}_n^*$ the multiplicative inverse such that $a \cdot a^{-1} \equiv 1 \mod n$
- To compute the multiplicative inverse of $a \in \mathbb{Z}_n^*$: use **EXT-EUCLID**(a, n) to get $ax + ny = 1$ or $ax \equiv 1 \mod n$

Finding the multiplicative inverse of $a \in \mathbb{Z}_n^*$

Modular division: given $x, y, n \in \mathbb{N}$, compute $x/y \mod n$ (if it exists!).

- As $(\mathbb{Z}_n^*, \cdot)$ is an abelian group, then $\forall a \in \mathbb{Z}_n^*, \exists a^{-1} \in \mathbb{Z}_n^*$ the multiplicative inverse such that $a \cdot a^{-1} \equiv 1 \mod n$
- To compute the multiplicative inverse of $a \in \mathbb{Z}_n^*$: use **EXT-EUCLID**(a, n) to get $ax + ny = 1$ or $ax \equiv 1 \mod n$
- Therefore, $x = a^{-1}$ and it can be computed in time $O(N^3)$.

Find the multiplicative inverse of $5 \mod 11$:

EXT-EUCLID $(5, 11) = (1, -2, 1) \Rightarrow 5 \cdot (-2) \equiv 1(\mod 11)$, and -2 is the multiplicative inverse of $5 \mod 11$. ($-2 = 9$ in $\mathbb{Z}_{11}^*$)

Find the multiplicative inverse of $21 \mod 91$

Notice $91 = 13 \cdot 7$ and $21 = 3 \cdot 7$ therefore $\gcd(91, 21) = 7 \Rightarrow 21$ does't have inverse $\mod 91$.

Chinese Remainder Theorem

- The Chinese Remainder Theorem provides a correspondence between a system of equations mod pairwise primes and an equation mod their product.

Chinese Remainder Theorem

- The Chinese Remainder Theorem provides a correspondence between a system of equations mod pairwise primes and an equation mod their product.
- Given groups G_1, G_2 their cartesian product $G_1 \times G_2 = \{(a_1, a_2) | a_1 \in G_1, a_2 \in G_2\}$.

  Multiplication: $(a_1, a_2) \cdot (b_1, b_2) = (a_1 \cdot b_1, a_2 \cdot b_2)$

Chinese Remainder Theorem

- The Chinese Remainder Theorem provides a correspondence between a system of equations mod pairwise primes and an equation mod their product.
- Given groups G_1, G_2 their cartesian product $G_1 \times G_2 = \{(a_1, a_2) | a_1 \in G_1, a_2 \in G_2\}$.

  Multiplication: $(a_1, a_2) \cdot (b_1, b_2) = (a_1 \cdot b_1, a_2 \cdot b_2)$
- If $n = q_1 q_2 \cdots q_k$, with $\gcd(q_i, q_j) = 1$, the CRT describes the structure of $(\mathbb{Z}_n, +_n, \cdot_n)$ as identical to $\mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2} \times \cdot \times \mathbb{Z}_{q_k}$, each with $+_{q_i}, \cdot_{q_i}$.

Chinese Remainder Theorem

- The Chinese Remainder Theorem provides a correspondence between a system of equations mod pairwise primes and an equation mod their product.
- Given groups G_1, G_2 their cartesian product $G_1 \times G_2 = \{(a_1, a_2) | a_1 \in G_1, a_2 \in G_2\}$.

  Multiplication: $(a_1, a_2) \cdot (b_1, b_2) = (a_1 \cdot b_1, a_2 \cdot b_2)$
- If $n = q_1 q_2 \cdots q_k$, with $\gcd(q_i, q_j) = 1$, the CRT describes the structure of $(\mathbb{Z}_n, +_n, \cdot_n)$ as identical to $\mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2} \times \cdot \times \mathbb{Z}_{q_k}$, each with $+_{q_i}, \cdot_{q_i}$.
- It has a many applications into algorithmics and cryptography, as working in $(\mathbb{Z}_{q_i}, +_{q_i}, \cdot_{q_i})$ could be more efficient than working in $(\mathbb{Z}_n, +_n, \cdot_n)$

Chinese Remainder Theorem

Lemma

If $\gcd(q_1, q_2) = 1$, *then* $\mathbb{Z}_{q_1 q_2} \cong \mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2}$.

Sketch Proof Define $f : \mathbb{Z}_{q_1 q_2} \to \mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2}$ by $f(a) = (a \mod q_1, a \mod q_2)$ ad prove that f is a bijection. $\square$

Chinese Remainder Theorem

Lemma

If $\gcd(q_1, q_2) = 1$, *then* $\mathbb{Z}_{q_1 q_2} \cong \mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2}$.

Sketch Proof Define $f : \mathbb{Z}_{q_1 q_2} \to \mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2}$ by $f(a) = (a \mod q_1, a \mod q_2)$ ad prove that f is a bijection. □

By using inductively the previous lemma:

Theorem (CRT, Sun Tzi Suan Ching, III), Euler XVIII)

Let $n = q_1 \cdot q_2 \cdot q_k$ *with* $\gcd(q_i, q_j) = 1$. *Then,*

$$\mathbb{Z}_n \cong \mathbb{Z}_{q_1} \times \cdots \times \mathbb{Z}_{q_k}.$$

Notice we must have $q_i = p_i^c$ for prime p_i and constant c.

Chinese Remainder Theorem

Corollary

Let $n = q_1 \cdot q_2 \cdot q_k$ *with* $\gcd(q_i, q_j) = 1$, *let* $r_1, \ldots, r_n$ *be integers s.t.* $0 \leq r_i < 0$, *for* $1 \leq i \leq k$. *Then, the system of simultaneous* k *equations:* $\{x \equiv r_i \mod q_i\}_{i=1}^{k}$, *has a unique solution.*

CRT: Example-1

The procedure to apply CRT:

1. For all i, $1 \leq i \leq k$:
   - $m_i = n/q_i$,
   - $c_i = m_i(m_i^{-1} \mod q_i)$,
2. $x \equiv \sum_{i=1}^{n} r_i c_i (\mod n)$.

CRT: Example-1

The procedure to apply CRT:

1. For all i, $1 \leq i \leq k$:
   - $m_i = n/q_i$,
   - $c_i = m_i(m_i^{-1} \mod q_i)$,
2. $x \equiv \sum_{i=1}^{n} r_i c_i (\mod n)$.

Sun Tzi Suan Ching's problem: *We have a number of things, but we do not know exactly how many. If we count them by threes we have two left over. If we count them by fives we have three left over. If we count them by sevens we have two left over. How many things are there?*

$$
\begin{aligned}
x &\equiv 2 \mod 3 \\
x &\equiv 3 \mod 5 \\
x &\equiv 2 \mod 7
\end{aligned}
$$

which yields $x \equiv 23 \mod 105$

CRT: Example-2

We want to do arithmetic on integers modulo $n = 8633$.
$n = 8633 = 89 \times 97 = q_1 q_2 \Rightarrow m_1 = 97, m_2 = 89$