SI120 Discussion 3

TA team

function (map)

$f: A \to B$ assigns a unique element $b \in B$ for all $a \in A$.

- **injective**: $f(a_1) = f(a_2) \Rightarrow a_1 = a_2$
- **surjective**: $f(A) = B$
- **bijective**: injective and surjective

- We say that A, B **have the same cardinality** ($|A| = |B|$) if there is a bijection $f : A \to B$
  - We say that $|A| \leq |B|$ if there exists an injection $f : A \to B$.
    - If $|A| \leq |B|$ and $|A| \neq |B|$, we say that $|A| < |B|$

THEOREM: (Cantor) Let A be any set. Then $|A| < |\mathcal{P}(A)|$.

1. (15 points) Let $a, b \in \mathbb{Z}$ with $a \geq b > 0$, and let $q = \lfloor a/b \rfloor$. Show that $\ell(a) - \ell(b) - 1 \leq \ell(q) \leq \ell(a) - \ell(b) + 1$, where $\ell(x)$ is the length of the binary representation of an integer x.

$$\forall x \in \mathbb{N}^*, 2^{l(x)-1} \leq x < 2^{l(x)}, l(x) = \lfloor \log_2 x \rfloor + 1$$

1.

$$2^{l(a)-1} \leq a < 2^{l(a)}, 2^{l(b)-1} \leq b < 2^{l(b)}$$

$$2^{l(a)-l(b)-1} < \frac{a}{b} < 2^{l(a)-l(b)+1}$$

$$2^{l(a)-l(b)-1} \leq \lfloor \frac{a}{b} \rfloor \leq 2^{l(a)-l(b)+1} - 1$$

讨论 l(a)-l(b)是否为 0，因为左边可能不是整数

$$l(a) - l(b) - 1 \leq l(q) \leq l(a) - l(b) + 1$$

2.

$$\lfloor A+B \rfloor \geq \lfloor A \rfloor + \lfloor B \rfloor$$

$$\lfloor A \rfloor - \lfloor B \rfloor - 1 \leq \lfloor A-B \rfloor \leq \lfloor A \rfloor - \lfloor B \rfloor$$

$$\lfloor \log_2(\frac{b}{a}) \rfloor = \lfloor \log_2(\lfloor \frac{b}{a} \rfloor) \rfloor$$

2. (25 points) Implement EEA (Extended Euclidean Algorithm).

ALGORITHM: compute $d = \gcd(a, b)$, s, t such that $as + bt = d$

- **Input**: $a, b \ (a \geq b > 0)$
- **Output**: $d = \gcd(a, b)$, integers s, t such that $d = as + bt$
  - $r_0 = a; r_1 = b; \binom{s_0}{t_0} = \binom{1}{0}; \binom{s_1}{t_1} = \binom{0}{1};$
  - $r_0 = r_1 q_1 + r_2 \ (0 < r_2 < r_1); \ \binom{s_2}{t_2} = \binom{s_0}{t_0} - q_1 \binom{s_1}{t_1}$
  - $\vdots$
  - $r_{i-1} = r_i q_i + r_{i+1} \ (0 < r_{i+1} < r_i); \binom{s_{i+1}}{t_{i+1}} = \binom{s_{i-1}}{t_{i-1}} - q_i \binom{s_i}{t_i}$
  - $\vdots$
  - $r_{k-2} = r_{k-1} q_{k-1} + r_k \ (0 < r_k < r_{k-1}); \binom{s_k}{t_k} = \binom{s_{k-2}}{t_{k-2}} - q_{k-1} \binom{s_{k-1}}{t_{k-1}}$
  - $r_{k-1} = r_k q_k$
  - output r_k, s_k, t_k

```python
def EEA(a,b):
    s0 = 1; t0 = 0; s1 = 0; t1 = 1
    while a%b !=  0:
        q = a//b
        s2 = s0-q*s1; t2 = t0-q*t1
        s0,t0 = s1,t1; s1,t1 = s2,t2
        a,b = b,a%b
    return s1,t1
```

s=52693465174047597579174064083061206575761398656935114430811
2435606950663069562377006384677413803445132609836259065451941
5480012670786924252819925030347117153620759789600840565013488
9458156325490296036336342644796958477425288398387518178265890
7006563057148373685234965973219732121971442442376472912705292
01589

t=-
4922435602557020575264036911319758978419249536244008420108775
7193437212741118960024592916678950802342924534115789543242617
9365107718666362589094840035084251285306016811645985979248393
7224361285850400246381718448690438802997126844191121984884459
0762141055813365169533361189741247565502362579257453658280613
873

3. (25 points) Implement the Square-and-Multiply algorithm.

ALGORITHM: compute $a^e \bmod n$ in polynomial time

- **Input:** $a \in \{0,1,\ldots,n-1\}$; $e = (e_{k-1} \cdots e_0)_2 \quad //k = \ell(e)$
  - $e = e_{k-1} \cdot 2^{k-1} + \cdots + e_1 \cdot 2^1 + e_0 \cdot 2^0$
- **Output:** $a^e \bmod n$
  - **Square**: this step requires $O(k)$ multiplications modulo n
    - $x_0 = a$
    - $x_1 = (x_0^2 \bmod n) = (a^2 \bmod n)$
    - $x_2 = (x_1^2 \bmod n) = (a^{2^2} \bmod n)$
    - $\ldots$
    - $x_{k-1} = (x_{k-2}^2 \bmod n) = (a^{2^{k-1}} \bmod n)$
  - **Multiply**: this step requires $O(k)$ multiplications modulo n
    - $(a^e \bmod n) = (x_0^{e_0} \cdot x_1^{e_1} \cdots x_{k-1}^{e_{k-1}} \bmod n)$

```python
def SAM(a,e,n):
    result = 1
    while e > 0:
        if e & 1:
            result = result * a % n
        a = (a * a) % n
        e = e >> 1
    return result
```

19489389945386041607071081817241920919542635233623116738469155055
20625915922643693886546508713351109692750915684157878314121214348
91999235290979965397926547335052787068125208309422099919003183364
35802408907249020763770922682237250909513951994814724102553142432
60591665020918693044381737199432444238061823906089977020969899711
34105963997915957273941960090533678167318836865046871071816483210
94994097671995305419040805120814031555590587098823477471474182303
58814131381147208291328747857991048977465984265721979324595417184
75031700171514407373804788401894603784580054764847429538488131703
745484558069777675820760128018344

4. (20 points) Solve the following linear congruence equations:

(1) $17x \equiv 11 (\text{mod} 23)$;

(2) $55x \equiv 35 (\text{mod} 75)$.

(1) $17X \equiv 11 \mod 23$

$d = \gcd(17, 23) = 1$ —— 2'

$t = (\frac{a}{d})^{-1} \mod (\frac{n}{d})$

$= (17)^{-1} \mod 23$ —— 4'

$= 19 \mod 23$ —— 7'

$X \equiv (\frac{b}{d})\, t \mod (\frac{n}{d})$

$\equiv 11 \times 19 \mod 23$ —— 9'

$\equiv 2 \mod 23$ —— 10'

(2) $55X \equiv 35 \mod 75$

$d = \gcd(55, 75) = 5$ —— 2'

$t = (\frac{a}{d})^{-1} \mod (\frac{n}{d})$

$= (11)^{-1} \mod 15$ —— 4'

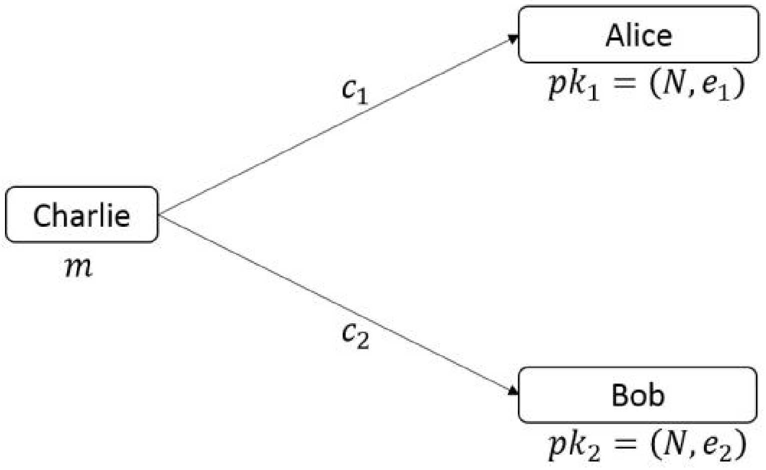
$= 11 \mod 15$ —— 7'

$X \equiv (\frac{b}{d}) \cdot t \mod (\frac{n}{d})$

$\equiv 7 \times 11 \mod 15$ —— 9'

$\equiv 2 \mod 15$ —— 10'

5. (15 points) See the following figure. Alice and Bob trust each other very much. They set their RSA public keys as $pk_1 = (N, e_1)$ and $pk_2 = (N, e_2)$, respectively. Charlie wants to send a private message m to Alice and Bob, where $0 \leq m < N$ is an integer and $\gcd(m, N) = 1$. To this end, Charlie encrypts m as $c_1 = m^{e_1} \bmod N$ and $c_2 = m^{e_2} \bmod N$; and then sends c_1 to Alice and sends c_2 to Bob.



Suppose that $\gcd(e_1, e_2) = 1$ and Eve sees all public keys and ciphertexts. Determine if Eve can learn the value of m.

Yes, Eve can learn the value of m.
According to the process of RSA, we have:

$$\begin{aligned} c_1 &= m^{e1}\ mod\ N \\ c_2 &= m^{e2}\ mod\ N \end{aligned}$$

We know that: $gcd(e_1, e_2) \ = \ 1$
By the Bezout's theorem:

$$\exists\ s, t \in\ \mathbb{Z}, s.t.\ e_1 * s + e_2 * t = 1$$

Where s,t can be found by EEA.

$$\begin{aligned} c_1^s * c_2^t\ mod\ N\ &= (m^{e1}\ mod\ N)^s * (m^{e2}\ mod\ N)^t \\ &= (m^{e1*s} * m^{e2*t})\ mod\ N \\ &= m^{e1*s+e2*t}\ mod\ N \end{aligned}$$

$$\because\ e_1 * s + e_2 * t = 1$$

$$\therefore\ c_1^s * c_2^t\ mod\ N\ =\ m\ mod\ N$$

$$\therefore\ c_1^s * c_2^t \equiv m\ mod\ N$$

Proved.