Modular Arithmetic

Week 6

Discrete Mathematics

Problem

The 24-hour clock numbers from 0 to 23.
Hanoi time zone: GMT +7
Hawai'i time zone: GMT -10

The time is 9:00 now in Hanoi. What is the time in Hawai'i?

9 – 17 = – 8 , but – 8 is not in the range from 0 to 23. What to do? Wrap around. The time in Hawai'i now is 16.

Problem

In the world of computer science, we work with finite sets.

Binary digit: 0 1
32-bit integers: from –2,147,483,648 to 2,147,483,647

What if the sum of two integers is greater than 2,147,483,647?

Answer: we wrap around to 0 every time we reach 2,147,483,647

Modular Arithmetic

The numbers used are limited to the range {0, 1, …, m-1}

m = 7

2 + 3 ≡ 5 (mod 7)	**7 divides (2 + 3 – 5)**
3 + 4 ≡ 0 (mod 7)	**(3 + 4 – 0) is divisible by 7**
4 + 6 ≡ 3 (mod 7)	**7 divides (4 + 6 – 3)**

4 - 2 ≡ 2 (mod 7)	**7 divides (4 - 2 – 2)**
3 - 4 ≡ 6 (mod 7)	**(3 - 4 – 6) is divisible by 7**

Modular Arithmetic

Theorem: If $a \equiv c \bmod m$ and $b \equiv d \bmod m$, then $a + b \equiv c + d \bmod m$ and $a \times b \equiv c \times d \bmod m$. which means that $a + b \equiv c + d \bmod m$.

Consider the expression $(13 + 11) \times 18 \quad \bmod 7$, using the theorem several times we have

$$\begin{aligned}
(13 + 11) \times 18 &\equiv (6 + 4) \times 4 \bmod 7 \\
&\equiv 10 \times 4 \bmod 7 \\
&\equiv 3 \times 4 \bmod 7 \\
&\equiv 12 \bmod 7 \\
&\equiv 5 \bmod 7
\end{aligned}$$

Greatest Common Divisor

```
int gcd(int x, int y)
{
        if (y = = 0) return (x)
        else return (gcd(y, x mod y))
}
```

```
gcd(8,6)                        // x = 8, y = 6
        gcd(6,2)                // 2 = 8 mod 6, x = 6, y = 2
                gcd(2,0)        // 0 = 6 mod 2
                return 2        // return x
        return 2                // return gcd(y, x mod y)
return 2                        // return gcd(y, x mod y)
```

Congruence

Solve the congruence $3x \equiv 4 \pmod{13}$

$$
\begin{aligned}
3x &\equiv 4 \pmod{13} \\
\Leftrightarrow 12x &\equiv 16 \pmod{13} \qquad (1)
\end{aligned}
$$

We have

$$13x \equiv 0 \pmod{13} \qquad (2)$$

(2)-(1) then,

$$
\begin{aligned}
x &\equiv -16 \pmod{13} \\
\Leftrightarrow x &\equiv -16 + 13 \times 2 \pmod{13} \\
\Leftrightarrow x &\equiv 10 \pmod{13}
\end{aligned}
$$

How can we solve the general problem? Where do the eq. (1), (2) come from?

Congruence

First, find a linear equation

$$\gcd(3,13) = 1 = 13.(1) + 3.(-4) \qquad \textbf{(*)}$$

Second, from $\quad 3x \equiv 4 \pmod{13}$

$$(-4).3.x \equiv (-4).4 \pmod{13} \qquad \textbf{(1)}$$

$$(1).13x \equiv 0 \pmod{13} \qquad \textbf{(2)}$$

(2)+(1) then,

$$\begin{aligned} x &\equiv -16 \pmod{13} \\ \Leftrightarrow x &\equiv -16 + 13 \times 2 \pmod{13} \\ \Leftrightarrow x &\equiv 10 \pmod{13} \end{aligned}$$

Congruence

Linear Congruence Theorem

If a and b are any integers, n is a positive integer, $d = \gcd(a, n)$ then the congruence

$$ax \equiv b \pmod{n}$$

has solution x if and only if b is divisible by d ($d|b$)

and the set of all solution is given by

$$\{x_0 + k.\frac{n}{d} \mid k \in \mathbb{Z}\}$$

where x_0 is one solution. How to find x_0?

$$\gcd(a,n) = n.s + a.r \qquad x_0 = r.b/d \pmod{n/d}$$

Find $3x \equiv 2 \pmod{6}$, then

e.g. has no solution

Euclidean Algorithm

Find x if $ax \equiv b \pmod{n}$

$$28x \equiv 8 \pmod{48}$$

Dividend	Divisor	Quotient	Remainder
48	28	1	20
28	20	1	8
20	8	2	4
8	4	2	0

Euclidean Algorithm

4=20-8.2	4 = 20.(1) + 8.(-2)
4=20-2(28-20.1)	
4=20.3-28.2	4 = 28.(-2) + 20.(3)
4=(48-28).3-28.2	
4=48.3-28.5	4 = 48.(3) + 28.(-5)

$$\gcd(48, 28) = 4$$

$$x_0 = -5.8/4 \pmod{48/4} = 2$$

$$\begin{aligned} x &= \{x_0 + k.\frac{n}{d}, \quad k = 0, 1, 2, ...\} \\ &= \{2, 14, 26, 38\} \end{aligned}$$

Multiplicative Inverse

y is called the multiplicative inverse of $x \pmod{m}$ if $xy \equiv 1 \pmod{m}$

2 is multiplicative inverse of 4 mod 7
because 2 x 4 ≡ 1 mod 7
5 is multiplicative inverse of 3 mod 7
because 5 x 3 ≡ 1 mod 7

0 has no multiplicative inverse mod 7
2 has no multiplicative inverse mod 6

Theorem

Let m, x be positive numbers such that $\gcd(m,x)=1$. Then x has a multiplicative inverse mod m, and it is unique.

Proof:

Consider $0x, 1x, \ldots, (m-1)x$. If there exists $0 \le a < b \le m-1$, such that $ax = bx \bmod m$. Then $ax - bx = 0 \bmod m$. $(a - b)x = 0 \bmod m$. Since $\gcd(x,m)=1$, $a - b$ is an integer multiple of m. This is not possible. Therefore, $0x, 1x, \ldots, (m-1)x$ are all distinct values mod m. $ax = 1 \bmod m$ for exactly one a.

Multiplicative Inverse

```
(d,a,b) e_gcd(x,y) {
        if (y = = 0) then return (x, 1, 0)
        else {
                (d, a, b) = e_gcd(y, x mod y)
                return (d, b, a - (x div y) * b)
        }
}
```

```
e_gcd(7, 3)                         // x = 7, y = 3
  (d,a,b)=e_gcd(3, 1)               // 1 = 7 mod 3, x = 3, y = 1
    (d,a,b)=e_gcd(1, 0)             // 0 = 3 mod 1, x = 1, y = 0
        return (1, 1, 0)   // d = 1, a = 1, b = 0
      return (1, 0, 1)   // d = 1, a = 0, b = 1
   return (1, 1, -2)
```

Multiplicative Inverse

Find the inverse of 29 in modulo 48

$$29t \equiv 1 \pmod{48}$$

Dividend	Divisor	Quotient	Remainder
48	29	1	19
29	19	1	10
19	10	1	9
10	9	1	1
9	**1**	9	0

$$\gcd(48, 29) = 1$$

Euclidean Algorithm

1=10-9	1 = 10.(1) + 9.(-1)
1=10-(19-10.1)	
1=10.2-19	1 = 19.(-1) + 10.(2)
1=(29-19.1).2-19	
1=29.2-19.3	1 = 29.(2) + 19.(-3)
1=29.2-(48-29.1).3	
1=29.**5**-48.3	1 = 48.(-3) + 29.(5)

$$t = 5$$

Multiplicative Inverse

$$q_i = \text{flip-updown}(\text{Quotient})$$
$$\text{dvd} = \text{flip-updown}(\text{Dividend})$$
$$\text{dvs} = \text{flip-updown}(\text{Divisor})$$

$$r_1 = 1,\ s_1 = -q_1;$$
$$r_i = s_{(i-1)};$$
$$s_i = r_{(i-1)} - q_i.s_{(i-1)}$$

Dividend	Divisor	Quotient	Remainder
48	29	1	19
29	19	1	10
19	10	1	9
10	9	1	1
9	1	9	0

dvd	r	dvs	s
10	1	9	-1
19	-1	10	2
29	2	19	-3
48	-3	29	5

Remove this row

Euclidean Algorithm

$$1 = \text{dvd}.(r) + \text{dvs}.(s)$$

1 = 10.(1) + 9.(-1)
1 = 19.(-1) + 10.(2)
1 = 29.(2) + 19.(-3)
1 = 48.(-3) + 29.(5)

Multiplicative Inverse

Find the inverse of 15 in modulo 26

$$15t \equiv 1 \pmod{26}$$

Dividend	Divisor	Quotient	Remainder
26	15	1	11
15	11	1	4
11	4	2	3
4	3	1	1
3	1	3	0

q_i

Extended Euclidean Algorithm

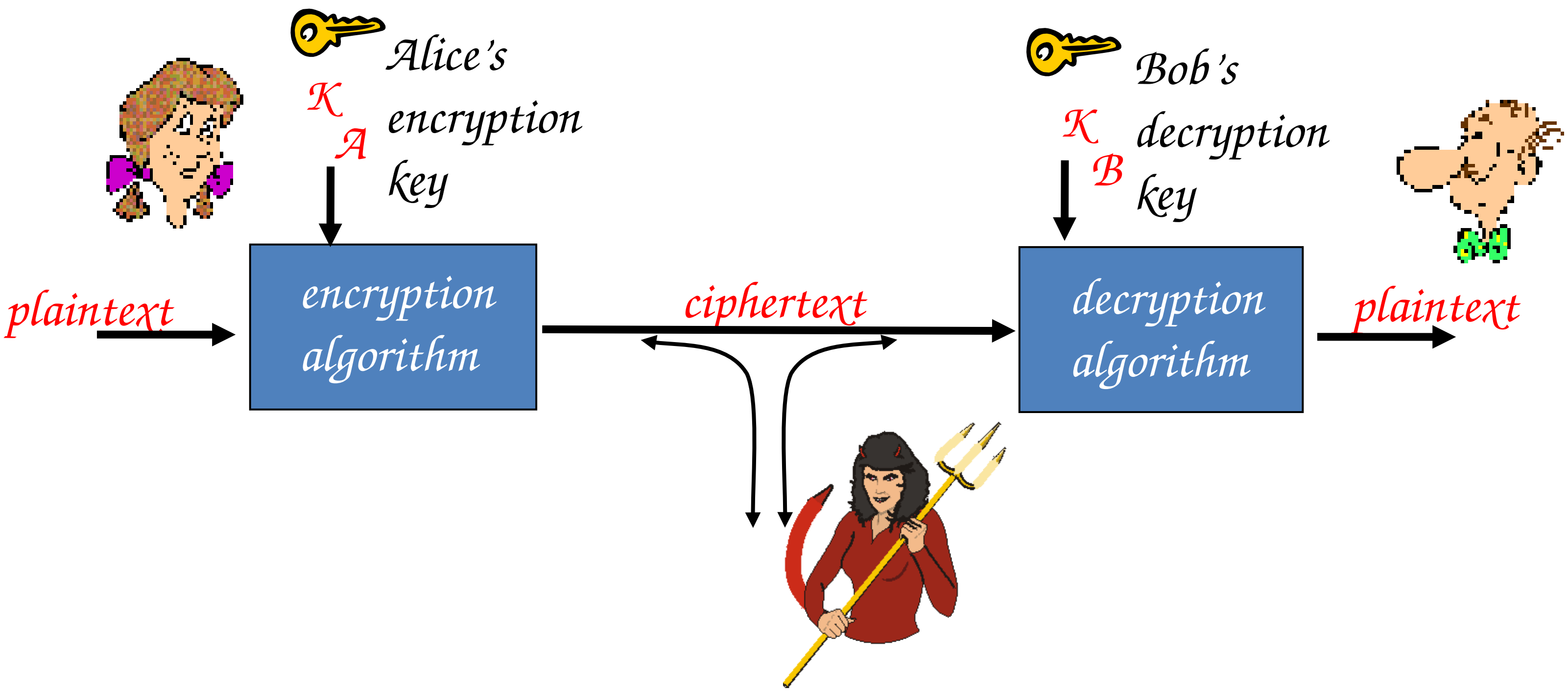
Step	
0	p0=0
1	p1=1
2	p2=0-1.(1) mod 26 = 25
3	p3=1-25.(1) mod 26 = 2
4	p4=25-2.(2) mod 26 = 21
	p5=2-21.(1) mod 26=-19 mod 26 = 7

$$t = 7$$

$$p_0 = 0; p_1 = 1; p_i = p_{i-2} - p_{i-1}.q_{i-2} \pmod{n}$$

$$15 \times 7 = 105 = 1 + 4 \times 26 \equiv 1 \pmod{26}$$

Cryptography



symmetric key crypto: sender, receiver keys *identical*

public-key crypto: encryption key *public*, decryption key *secret* (private)

Symmetric Key Cryptography

substitution cipher: substituting one thing for another

– monoalphabetic cipher: substitute one letter for another

```
 plaintext:  abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

E.g.:

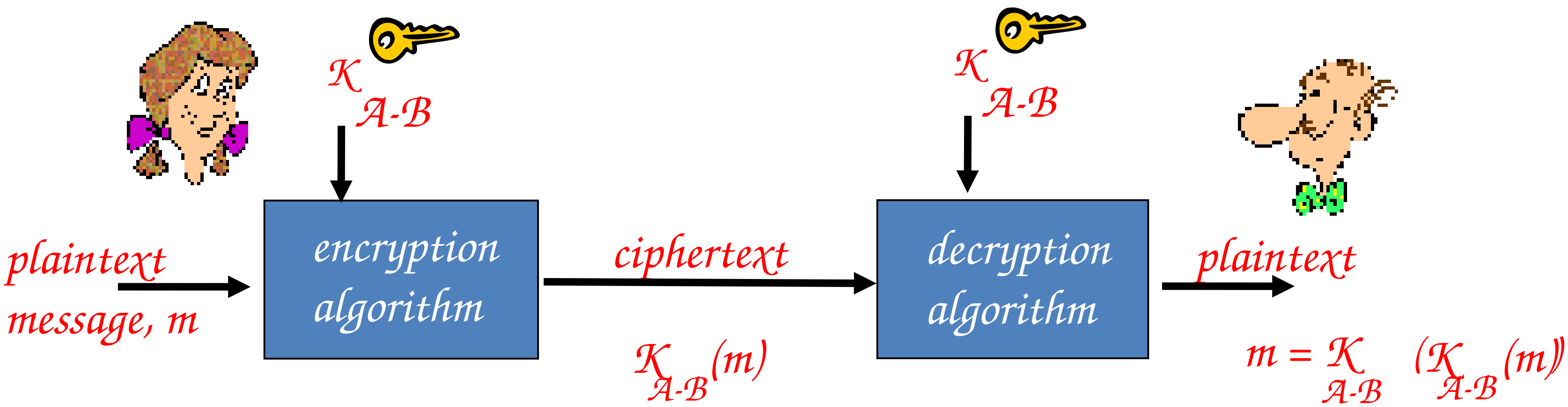
```
 Plaintext: bob. i love you. alice

ciphertext: nkn. s gktc wky. mgsbc
```

Q: How hard to break this simple cipher?:

- *brute force (how hard?)*
- *other?*

Symmetric Key Cryptography



symmetric key crypto: Bob and Alice share know same (symmetric) key: $K_{A\text{-}B}$

- e.g., key is knowing substitution pattern in mono alphabetic substitution cipher
- <u>Q:</u> how do Bob and Alice agree on key value?

Symmetric Key Crypto.: DES

DES: Data Encryption Standard

- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64-bit plaintext input
- How secure is DES?
  - DES Challenge: 56-bit-key-encrypted phrase ("Strong cryptography makes the world a safer place") decrypted (brute force) in 4 months
  - no known "backdoor" decryption approach
- making DES more secure:
  - use three keys sequentially (3-DES) on each datum
  - use cipher-block chaining

Symmetric Key Crypto.: AES

AES: Advanced Encryption Standard

- new (Nov. 2001) symmetric-key NIST standard, replacing DES
- processes data in 128 bit blocks
- 128, 192, or 256 bit keys
- brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

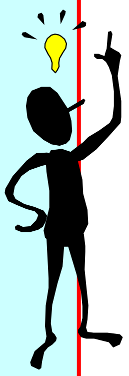
Public Key Cryptography

<u>*symmetric key crypto*</u>

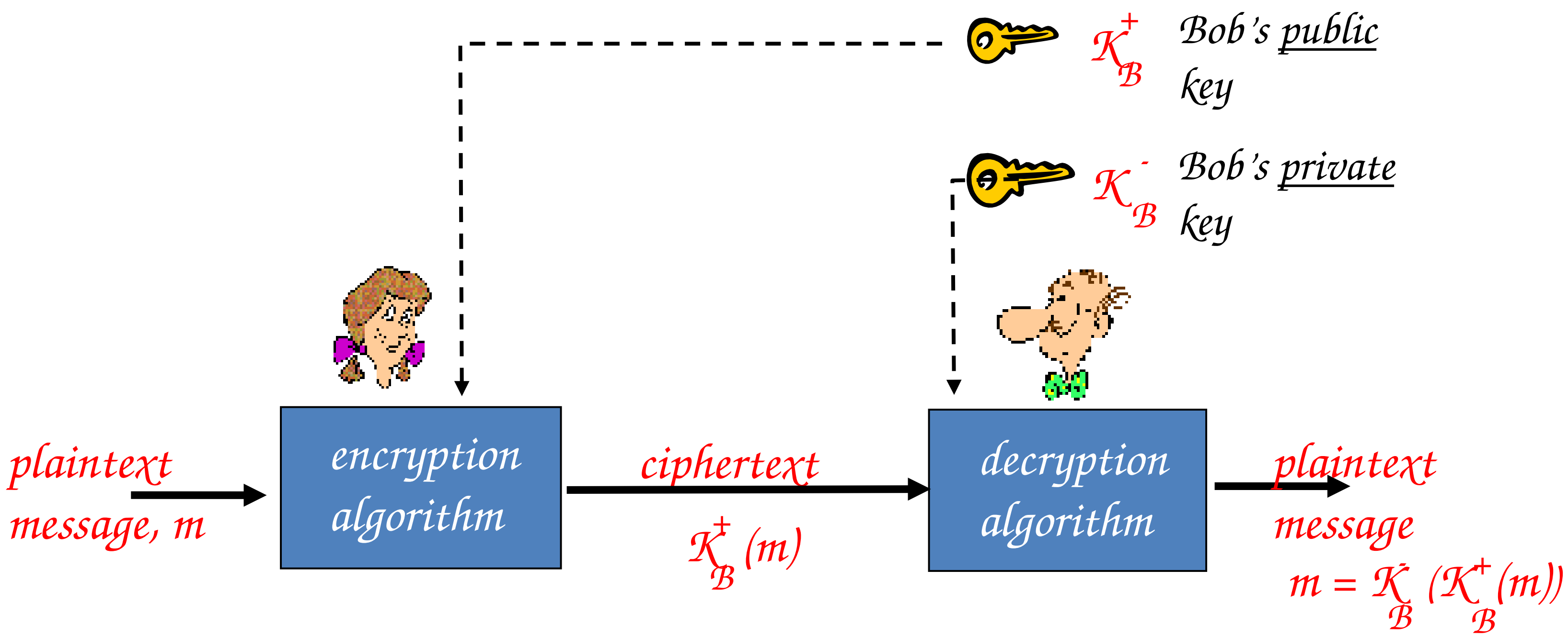
- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

<u>*public key cryptography*</u>

- *radically different approach [Diffie-Hellman76, RSA78]*
- *sender, receiver do not share secret key*
- *public encryption key known to all*
- *private decryption key known only to receiver*



Public Key Cryptography



Public Key Encryption Algorithm

Requirements:

1. need $K_B^+(\cdot)$ and $K_B^-(\cdot)$ such that

$$K_B^-(K_B^+(m)) = m$$

2. *given public key K_B^+, it should be impossible to compute private key K_B^-*

RSA: Rivest, Shamir, Adelson algorithm

Asymmetric Key: RSA

- Choose two large random prime numbers p and q
- Compute $n = pq$, n is used as the modulus for both the public and private keys.
- Compute $z = (p-1)(q-1)$
- Choose an integer e such that $1 < e < z$, and $\gcd(e,z) = 1$ (e, z are relatively prime). (n,e) is released as the public key.
- Compute d as a multiplicative inverse of e modulo z, i.e. $ed \bmod z=1$. (n,d) is kept as the private key.

RSA: Encryption, Decryption

0. *Given (n,e) and (n,d) as computed above*

1. *To encrypt bit pattern, m, compute*

$$c = m^e \bmod n$$

(i.e., remainder when m^e *is divided by* n *)*

2. *To decrypt received bit pattern, c, compute*

$$m = c^d \bmod n$$

(i.e., remainder when c^d *is divided by* n *)*

Magic happens!

$$m = \underbrace{(m^e \bmod n)^d}_{} \bmod n$$

$$m = c^d \bmod n$$

RSA: Example

Bob chooses p=5, q=7. Then n=35, z=24.

e=5 (so e, z relatively prime).
d=29 (so ed-1 exactly divisible by z).

	letter	m	m^e	$c = m^e \bmod n$
encrypt:	l	12	248832	17

	c	c^d	$m = c^d \bmod n$	letter
decrypt:	17	481968572106750915091411825223071697	12	l

RSA: Why $m = (m^e \bmod n)^d \bmod n$

Useful number theory result: If p,q prime and $n = pq$, then:

$$x^y \bmod n = x^{y \bmod (p-1)(q-1)} \bmod n$$

$$(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$$

$$= m^{ed \bmod (p-1)(q-1)} \bmod n$$

(using number theory result above)

$$= m^1 \bmod n$$

(since we chose ed to be divisible by $(p-1)(q-1)$ with remainder 1)

$$= m$$

RSA:Another Important Property

The following property will be *very* useful later:

$$\underbrace{K_B^-(K_B^+(m))}_{\text{use public key first, followed by private key}} = m = \underbrace{K_B^+(K_B^-(m))}_{\text{use private key first, followed by public key}}$$

Result is the same!