COMP170
Discrete Mathematical Tools for Computer Science

Intro to Crypto and Mod

Version 2.0: Last updated, May 13, 2007

Discrete Math for Computer Science
K. Bogart, C. Stein and R.L. Drysdale
Section 2.1, pp. 43-54

Godfrey Harold Hardy

b. 1877. d. 1947

was a prominent British mathematician, known for his achievements in number theory and mathematical analysis.

from *http://en.wikipedia.org/wiki/G._H._Hardy*

In his 1940 autobiography *A Mathematician's Apology*, Hardy wrote

"The great modern achievments of applied mathematics have been in relativity and quantum mechanics, and these subjects are, at present, **almost as 'useless' as the theory of numbers."**

and

"... then the great bulk of higher mathematics is **useless.** *Modern Geometry and algebra,* **the theory of numbers**, *the theory of aggregates and functions, relativity, quantum mechanics – no one of them stands the test much better than another, . . ."*

If he could see the world now, G.H. Hardy would be spinning in his grave.

Number theory, introduced in this lecture, is the basis of modern coding theory.
Computer security and e-commerce would be *impossible* without it.

relativity and quantum theory turned out to be pretty useful as well . . ..

At one point, not long ago, the largest employer of mathematicians in the United States, and therefore probably the world, was the National Security Agency (NSA). The NSA is the largest spy agency in the US – bigger than the CIA – and has the responsibility for code design and breaking.

(Euclid's Division Theorem)
Let n be a *positive* integer. Then for *every* integer m, there exist *unique* integers q and r such that $m = nq + r$ and $0 \leq r < n$.

This will be proven in next lecture.
It says that $m \bmod n$ is *uniquely* defined.

2.1 Cryptography and Modular Arithmetic

- Arithmetic Modulo n
- Introduction to Cryptography
- Private-Key Cryptography
  - Caesar Ciphers: Cryptography Using Addition $\bmod n$
  - Cryptography Using Multiplication $\bmod n$
- Public-Key Cryptography

A quick review of the laws of arithmetic over the real numbers

- The *commutative laws* for addition and multiplication

  $a + b = b + a; \quad ab = ba$

  Ex: $3 + 7.2 = 7.2 + 3; \quad 3 \cdot 5 = 5 \cdot 3.$

- The *associative laws* for addition and multiplication

  $a + (b + c) = (a + b) + c; \quad a(bc) = (ab)c$

  Ex: $5 + (3 + 7) = (5 + 3) + 7; \quad 5 \cdot (3 \cdot 7) = (5 \cdot 3) \cdot 7$

- The *distributive law*

  $(a + b)c = ac + bc$

  $(5 + 3) \cdot 7 = 5 \cdot 7 + 3 \cdot 7$

- Every number a has an *additive inverse* $-a$ such that

  $a + (-a) = 0.$ Ex: $5 + (-5) = 0.$

- Every number $a \neq 0$ has a *multiplicative inverse* a^{-1} s.t.

  $aa^{-1} = 1.$ Ex: $5 \cdot \frac{1}{5} = 1.$

2.1 Cryptography and Modular Arithmetic

- Arithmetic Modulo n

- Introduction to Cryptography
- Private-Key Cryptography
  - Caesar Ciphers: Cryptography Using Addition $\bmod n$
  - Cryptography Using Multiplication $\bmod n$

- Public-Key Cryptography

Modular Arithmetic

Definition (1st version);
Let m, n be positive integers. Then $m \bmod n$ is the remainder left when dividing m by n.

Ex: $25 \bmod 4 = 1; \quad 25 \bmod 5 = 0; \quad 27 \bmod 5 = 2$.

We would like to extend this definition to *negative* m as well.

Definition (2nd version);
For an integer m and positive integer n,
$m \bmod n$ is the smallest nonnegative integer r such that, for some integer q, $m = nq + r$.

Ex: $25 \bmod 4 = 1; \quad -25 \bmod 4 = 3$.

Definition (2nd version);
For an integer m and positive integer n,
$m \bmod n$ is the smallest nonnegative integer r such that,
for some integer q, $m = nq + r$.

$25 \bmod 4 = 1$ because $25 = 4 \cdot 6 + 1$ and any other way of writing $25 = 4 \cdot q + r$ would have an r bigger than 1.

$-25 \bmod 4 = 3$ because $-25 = 4 \cdot (-7) + 3$ and any other way of writing $-25 = 4 \cdot q + r$ would have an r bigger than 3. (why?)

Note: In general, except if $[m \bmod n] = 0$,

$[(-m) \bmod n] = n - [m \bmod n]$ so

$[(-m) \bmod n] \neq [m \bmod n]$ unless

$[m \bmod n] = n/2$.

Arithmetic Modulo n

Compute

$21 \bmod 9$

$38 \bmod 9$

$(21 \cdot 38) \bmod 9$

$(21 \bmod 9) \cdot (38 \bmod 9)$

$(21 + 38) \bmod 9$

$(21 \bmod 9) + (38 \bmod 9)$

It looks as if $[(ab) \bmod n] = [(a \bmod n) \cdot (b \bmod n)]$
and $[(a+b) \bmod n] = [(a \bmod n) + (b \bmod n)]$

Is this true? **No!** Try $a = 2$, $b = 8$, $n = 9$.

So what is happening here?

True or false?

$i \bmod n = (i + 2n) \bmod n$? $\quad$ $i \bmod n = (i - 3n) \bmod n$?

Both true, since adding multiples of n to i does not change the value of the *remainder*, $i \bmod n$.

Lemma 2.2

$i \bmod n = (i + kn) \bmod n$ for all integers k.

Proof:

- By Euclid's Division Theorem, $i = nq + r$ (*), for *unique* integers q and r, with $0 \le r < n$.
- By (*) and definition of mod, $r = i \bmod n$.
- Adding kn to both sides, $i + kn = n(q + k) + r$ (**).
- From (**), Eucid's div thm and definition of mod, $r = (i + kn) \bmod n$, and we are done.

Lemma 2.3

$$\begin{aligned}(i+j) \bmod n &= (i+(j \bmod n)) \bmod n \\ &= ((i \bmod n)+j) \bmod n \\ &= ((i \bmod n)+(j \bmod n)) \bmod n,\end{aligned}$$

$$\begin{aligned}(i \cdot j) \bmod n &= (i \cdot (j \bmod n)) \bmod n \\ &= ((i \bmod n) \cdot j) \bmod n \\ &= ((i \bmod n) \cdot (j \bmod n)) \bmod n.\end{aligned}$$

Lemma 2.3

$$
\begin{aligned}
(i+j) \bmod n &= (i+(j \bmod n)) \bmod n \\
&= ((i \bmod n)+j) \bmod n \\
&= ((i \bmod n)+(j \bmod n)) \bmod n,
\end{aligned}
$$

Proof:

We prove that item on left is equal to bottom item on right. Proofs of all other equalities are very similar.

By *Euclid's Division Theorem*, for *unique* q_1 and q_2,

$i = (i \bmod n) + q_1 n$ and $j = (j \bmod n) + q_2 n$.

Adding these 2 equations together $\bmod n$ and using Lemma 2.2,

$$
\begin{aligned}
(i+j) \bmod n &= ((i \bmod n) + q_1 n + (j \bmod n) + q_2 n) \bmod n \\
&= ((i \bmod n) + (j \bmod n) + n(q_1+q_2)) \bmod n \\
&= ((i \bmod n) + (j \bmod n)) \bmod n.
\end{aligned}
$$

Definition:
Z_n is the set of integers $\{0, 1, \ldots, n-1\}$ with
addition $\mathrm{mod} n$ $\quad i +_n j = (i + j) \bmod n$ and
multiplication $\mathrm{mod} n$ $\quad i \cdot_n j = (i \cdot j) \bmod n$

- If $x \in Z_n$, then x is a variable with possible integral values between 0 and $n - 1$.
- If $x, y \in Z_n$, we use $x +_n y$ and $x \cdot_n y$ to perform algebraic operations on x, y.
- **Additive identity property**: $0 +_n i = i$.
  Multiplicative identity property: $1 \cdot_n i = i$.
- $a -_n b$ denotes $a +_n (-b)$.

Theorem 2.4
Addition and multiplication $\mod n$ satisfy the **commutative**, **associative** and **distributive** laws.

Proof: Commutativity of $+_n$ and $\cdot_n$ follows immediately from commutativity of ordinary addition and multiplication. We prove the associative law for addition in the following equations; the other laws follow similarly.

Theorem 2.4
Addition and multiplication $\bmod n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$a +_n (b +_n c) = (a + (b +_n c)) \text{ mod } n$$

$i +_n j = (i + j)$ mod n and $i \cdot_n j = (i \cdot j)$ mod n.

Theorem 2.4
Addition and multiplication $\mathrm{mod}\, n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$\begin{aligned} a +_n (b +_n c) &= (a + (b +_n c)) \text{ mod } n \\ &= (a + ((b + c) \text{ mod } n)) \text{ mod } n \end{aligned}$$

$i +_n j = (i + j) \text{ mod } n$ and $i \cdot_n j = (i \cdot j) \text{ mod } n$.

Theorem 2.4
Addition and multiplication $\mathrm{mod}\, n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$\begin{aligned} a +_n (b +_n c) &= (a + (b +_n c)) \text{ mod } n \\ &= (a + ((b + c) \text{ mod } n)) \text{ mod } n \\ &= (a + (b + c)) \text{ mod } n \end{aligned}$$

Lemma 2.3

Theorem 2.4
Addition and multiplication $\mathrm{mod}\, n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$\begin{aligned} a +_n (b +_n c) &= (a + (b +_n c)) \text{ mod } n \\ &= (a + ((b + c) \text{ mod } n)) \text{ mod } n \\ &= (a + (b + c)) \text{ mod } n \\ &= ((a + b) + c) \text{ mod } n \end{aligned}$$

Associative law for ordinary sums.

Theorem 2.4

Addition and multiplication $\mathrm{mod}\, n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$
\begin{aligned}
a +_n (b +_n c) &= (a + (b +_n c)) \text{ mod } n \\
&= (a + ((b + c) \text{ mod } n)) \text{ mod } n \\
&= (a + (b + c)) \text{ mod } n \\
&= ((a + b) + c) \text{ mod } n \\
&= ((a + b) \text{ mod } n + c) \text{ mod } n
\end{aligned}
$$

Lemma 2.3

Theorem 2.4
Addition and multiplication $\mathrm{mod}\, n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$\begin{aligned} a +_n (b +_n c) &= (a + (b +_n c)) \text{ mod } n \\ &= (a + ((b + c) \text{ mod } n)) \text{ mod } n \\ &= (a + (b + c)) \text{ mod } n \\ &= ((a + b) + c) \text{ mod } n \\ &= ((a + b) \text{ mod } n + c) \text{ mod } n \\ &= ((a +_n b) + c) \text{ mod } n \end{aligned}$$

$i +_n j = (i + j)$ mod n and $i \cdot_n j = (i \cdot j)$ mod n.

Theorem 2.4
Addition and multiplication $\mathrm{mod}\, n$ satisfy the **commutative**, **associative** and **distributive** laws.

$$
\begin{aligned}
a +_n (b +_n c) &= (a + (b +_n c)) \text{ mod } n \\
&= (a + ((b + c) \text{ mod } n)) \text{ mod } n \\
&= (a + (b + c)) \text{ mod } n \\
&= ((a + b) + c) \text{ mod } n \\
&= ((a + b) \text{ mod } n + c) \text{ mod } n \\
&= ((a +_n b) + c) \text{ mod } n \\
&= (a +_n b) +_n c
\end{aligned}
$$

$i +_n j = (i + j) \text{ mod } n$ and $i \cdot_n j = (i \cdot j) \text{ mod } n$.

2.1 Cryptography and Modular Arithmetic

- Arithmetic Modulo n
- Introduction to Cryptography
- Private-Key Cryptography
  - Caesar Ciphers: Cryptography Using Addition $\bmod n$
  - Cryptography Using Multiplication $\bmod n$
- Public-Key Cryptography

Introduction to Cryptography

Cryptography is the study of methods for sending and receiving secret messages.

A **sender** is trying to send a message to a **receiver**. An **adversary** wants to steal the message.

Sensitive information is **encrypted** — modified in such a way that it should be only understandable by the receiver and **undecipherable** to the adversary.

The method is deemed **successful** if the sender is able to communicate a messsage to the receiver without the adversary learning what that message was.

A *difficult* goal!

2.1 Cryptography and Modular Arithmetic

- Arithmetic Modulo n

- Introduction to Cryptography
- Private-Key Cryptography
  - Caesar Ciphers: Cryptography Using Addition $\bmod n$
  - Cryptography Using Multiplication $\bmod n$

- Public-Key Cryptography

Private-Key Cryptography

Sender and receiver agree in advance on a secret code and then send messages using that code.

Caesar cipher: A private-key cryptosystem in which letters of the alphabet are shifted (circularly) by some fixed amount.

This cipher is named after the Roman emperor **Julius Caesar** (b. 100BC, d. 44BC). Caesar supposedly used this type of cipher (with a shift of 3) to communicate with his generals.

Private-Key Cryptography

Sender and receiver agree in advance on a secret code and then send messages using that code.

Caesar cipher: A private-key cryptosystem in which letters of the alphabet are shifted (circularly) by some fixed amount.

Original message is called **plaintext** and the **encoded** text is called **ciphertext**.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D

Plaintext message: `ONE IF BY LAND AND TWO IF BY SEA.`

Ciphertext: `SRI MJ FC PERH ERH XAS MJ FC WIE.`

Easy to implement using arithmetic mod 26.

Use 0 for $\mathtt{A}$, 1 for $\mathtt{B}$, $\ldots$.
Convert a message to a sequence of numbers.
For example, $\mathtt{SEA}$ becomes $18\ 4\ 0$.

Q: What does the numerical representation of this word become if we shift every letter 2 places to the right?

A: Replace n by $(n+2) \bmod 26$, $\Rightarrow$ $\mathtt{SEA}$ becomes $20\ 6\ 2$.

Q: What if we shift every letter 13 places to the right?

A: Replace n by $(n+13) \bmod 26$, $\Rightarrow$ $\mathtt{SEA}$ becomes $5\ 17\ 13$.

A Caesar cipher with shift s can easily be implemented on most computers by replacing each "letter" n with $(n+s) \bmod 26$. Most computer languages can easily convert between text and numbers, and provide predefined mod functions.

A Caesar cipher with shift s can easily be implemented on most computers by replacing each "letter" n with $(n+s) \bmod 26$.

- E.G. To shift each letter 2 to the right, replace n by $n' = (n+2) \bmod 26$, $\quad\Rightarrow\quad$ `SEA` becomes $20\ 6\ 2$.

- How should the receiver decipher (decrypt) the message?

  Easy: Replace n' by $(n'-s) \bmod 26$,

  E.G. If $s=2$, then a received $20\ 6\ 2$ becomes $18\ 4\ 0$ which is SEA.

- So, we've just seen how $+_n$ on Z_n (for $n=26$) can be used to implement **encrypting** and **decrypting** Caesar ciphers.

A slightly different view

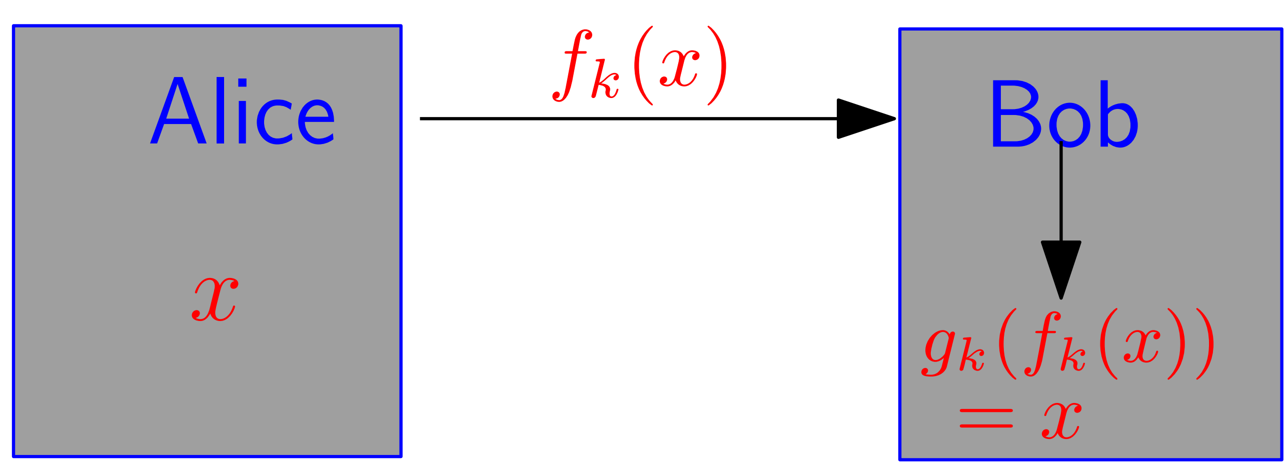
- A Caesar cipher has a private-key k
- To encode x, use the function $f_k(x) = x +_{26} k$
- To decode y, use the function $g_k(y) = y -_{26} k$
- Note that $g_k(y) = f_k^{-1}(y)$,
  i.e., $g(f(x)) = x$ and $f(g(y)) = y$

0) Bob & Alice know k

i) Alice has letter x

ii) She sends $f_k(x)$ to Bob

iii) Bob calculates
$x = g_k(f_k(x))$



2.1 Cryptography and Modular Arithmetic

- Arithmetic Modulo n
- Introduction to Cryptography
- Private-Key Cryptography
  - Caesar Ciphers: Cryptography Using Addition $\bmod n$
  - Cryptography Using Multiplication $\bmod n$
- Public-Key Cryptography

Cryptography Using Multiplication $\bmod n$

- We just saw how to use modular addition/subtraction to **encrypt/decrypt**. Now we'll discuss how to use modular multiplication. We assume that message is a *number*, M.

- Private key is some a, n.
  Encrypt: $f_{a,n}(M) = a \cdot M \bmod n = a \cdot_n M$
  Decrypt: "Divide" $f(M)$ by a mod n.

- In order for this to work *division* must exist and define an inverse to multiplication., i.e.

$$g_{a,n}(f_{a,n}(M)) = M \quad \text{where} \quad g_{a,n}(X) = a^{-1} \cdot_n X$$

Does division exist?

What exactly does division $\mathrm{mod}\, n$ mean?

Suppose, for some x, we had calculated $f(x) = a \cdot_n x$.
Does f^{-1} exist?

Consider the following three cases of a, x, n.

(a) $(a, x, n) = (4, 3, 12)$,
(b) $(a, x, n) = (3, 6, 12)$,
(c) $(a, x, n) = (5, 7, 12)$

Z_{12}	$\boxed{0}$	1	2	x $\boxed{3}$	4	5	$\boxed{6}$	7	8	$\boxed{9}$	10	11
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11
2	0	2	4	6	8	10	0	2	4	6	8	10
3	0	3	6	9	0	3	6	9	0	3	6	9
a 4	0	4	8	0	4	8	0	4	8	0	4	8
5	0	5	10	3	8	1	6	11	4	9	2	7
6	0	6	0	6	0	6	0	6	0	6	0	6
7	0	7	2	9	4	11	6	1	8	3	10	5
8	0	8	4	0	8	4	0	8	4	0	8	4
9	0	9	6	3	0	9	6	3	0	9	6	3
10	0	10	8	6	4	2	0	10	8	6	4	2
11	0	11	10	9	8	7	6	5	4	3	2	1

$f(x) = a \cdot_n x$

(a) $(a, x, n) = (4, 3, 12)$:

You send the message $f(x) = 4 \cdot_{12} 3 = 0$.

Recipient receives 0.

Problem:
There are 4 values of x, $(0, 3, 6, 9)$, s. t. $a \cdot_{12} x = 0$.

Recipient doesn't know which x you intended.

$\Rightarrow$ f^{-1} doesn't exist! Impossible to decrypt!

								x					
	Z_{12}	0	1	$\boxed{2}$	3	4	5	$\boxed{6}$	7	8	9	$\boxed{10}$	11
	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	1	2	3	4	5	6	7	8	9	10	11
	2	0	2	4	6	8	10	0	2	4	6	8	10
a	3	0	3	6	9	0	3	6	9	0	3	6	9
	4	0	4	8	0	4	8	0	4	8	0	4	8
	5	0	5	10	3	8	1	6	11	4	9	2	7
	6	0	6	0	6	0	6	0	6	0	6	0	6
	7	0	7	2	9	4	11	6	1	8	3	10	5
	8	0	8	4	0	8	4	0	8	4	0	8	4
	9	0	9	6	3	0	9	6	3	0	9	6	3
	10	0	10	8	6	4	2	0	10	8	6	4	2
	11	0	11	10	9	8	7	6	5	4	3	2	1

$$f(x) = a \cdot_n x$$

(b) $(a, x, n) = (3, 6, 12)$:

You send the message $f(x) = 3 \cdot_{12} 6 = 6$.

Recipient receives 6.

Problem:
There are 3 values of x, $(2, 6, 10)$, s. t. $a \cdot_{12} x = 6$.

Recipient doesn't know which x you intended.

$\Rightarrow$ f^{-1} doesn't exist! Impossible to decrypt!

Z_{12}	0	1	2	3	4	5	6	$\overset{x}{\boxed{7}}$	8	9	10	11
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11
2	0	2	4	6	8	10	0	2	4	6	8	10
3	0	3	6	9	0	3	6	9	0	3	6	9
4	0	4	8	0	4	8	0	4	8	0	4	8
a 5	0	5	10	3	8	1	6	11	4	9	2	7
6	0	6	0	6	0	6	0	6	0	6	0	6
7	0	7	2	9	4	11	6	1	8	3	10	5
8	0	8	4	0	8	4	0	8	4	0	8	4
9	0	9	6	3	0	9	6	3	0	9	6	3
10	0	10	8	6	4	2	0	10	8	6	4	2
11	0	11	10	9	8	7	6	5	4	3	2	1

$f(x) = a \cdot_n x$

(c) $(a, x, n) = (5, 7, 12)$:

You send the message
$f(x) = 5 \cdot_{12} 7 = 11$.

Recipient receives 11.

In fact,
7 is unique solution to
$5 \cdot_{12} x = 11$!

Recipient does know
which x you intended.

$\Rightarrow$ Recepient could decrypt this messsage!

What exactly does division $\bmod n$ mean?

Suppose, for some x, we had calculated $f(x) = a \cdot_n x$.
Does f^{-1} exist?

Consider the following three cases of a, x, n.
(a) $(a, x, n) = (4, 3, 12)$, (b) $(a, x, n) = (3, 6, 12)$,
(c) $(a, x, n) = (5, 7, 12)$

- We just saw that in cases (a) and (b), there is an $x' \neq x$ s.t. $f(x') = f(x)$ so recipient would not be able to decrypt message. This means that we can not use this $f(x)$ as an encoding function
- In case (c), given x, recipient could uniquely calculate x so $f(x)$ might be a good encoding function.
- $f(x)$ can be used as an encoding function
  when $f(x)$ has an inverse!

When does $f_{a,n}(x) = a \cdot_n x$ have an inverse?

$f_{a,n}(x) = a \cdot_n x$ has an inverse if and only if a and n are relatively prime, i.e., they have no common factors greater than 1.

In the next lecture we will see what this means and how to use it to define divsion in Z_n.

2.1 Cryptography and Modular Arithmetic

- Arithmetic Modulo n

- Introduction to Cryptography
- Private-Key Cryptography
  - Caesar Ciphers: Cryptography Using Addition $\bmod n$
  - Cryptography Using Multiplication $\bmod n$

- Public-Key Cryptography

- The Caesar cipher is a very simple form of private-key encryption. The *key* is the shift.
- There are many other more sophisticated and secure types of private-key encryption, see, e.g., COMP364
- The security of **all** of them rely on the fact that the *codebook* is kept secret between the sender and receiver.
- What happens if codebook is lost or stolen?
  No good answer to this
  How can we distribute different codebooks
  to millions of customers?
  Problem in e-commerce where each customer
  would need own codebook
- Motivation for Public-Key Cryptography

Public-Key Cryptosystems

- In **private-key cryptosystems** the sender and receiver *share* a private-key or codebook.

  The same key is used for encypting and decrypting.
  Implicit assumption: knowing how a message is encypted implies knowing how to decrypt it

- In **public-key cryptography** this is no longer true.

  Everybody has two keys; a **public key** and a **secret key**.

- **My public key:** Known by all. Used to send me a message
  My secret key: Known only by me.

  Used to decrypt messages sent to me that were encrypted using my public key.

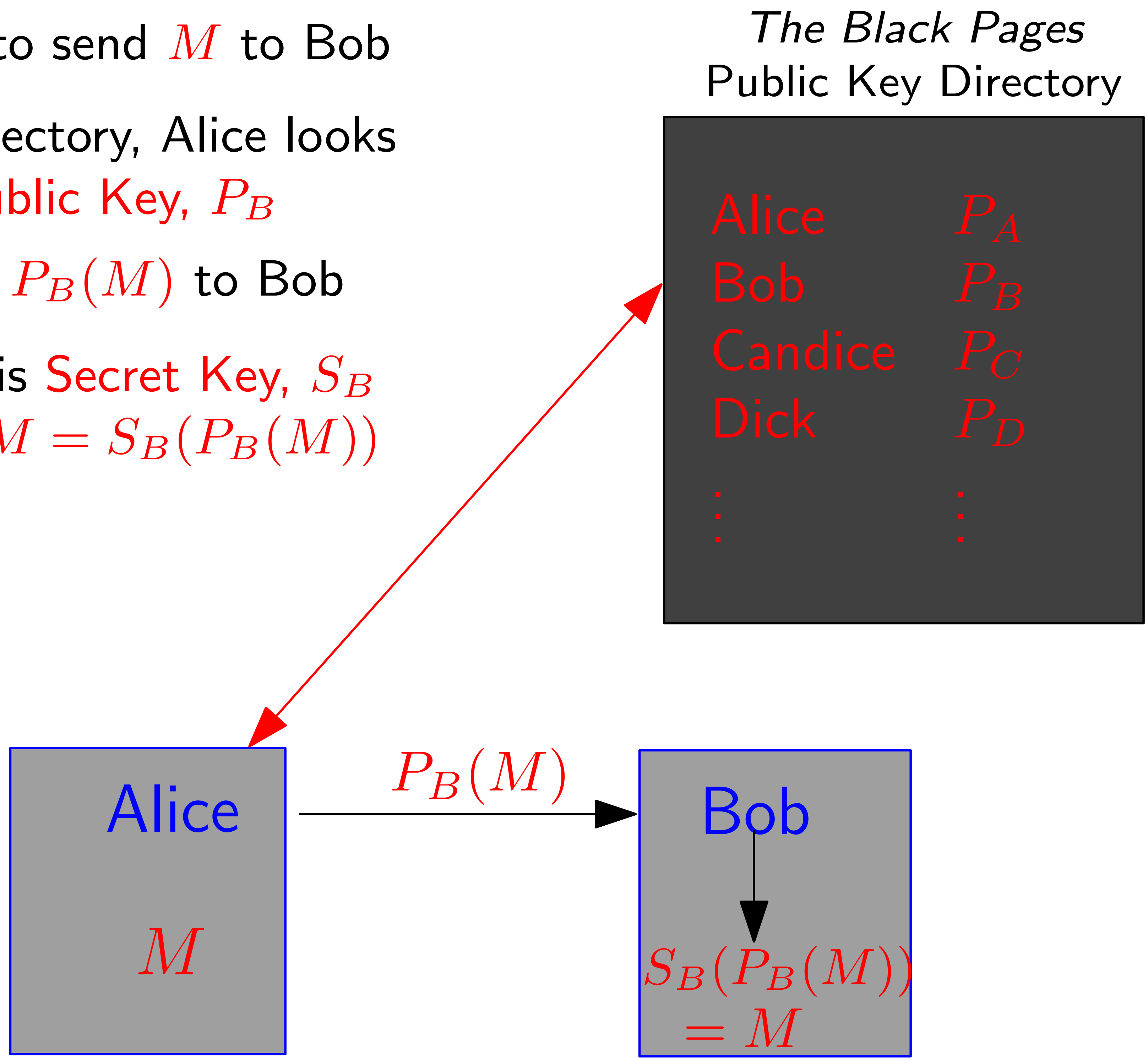
- **Important:** Even though everyone knows how messages sent to me were encrypted, only I can decrypt them. **!!??**

i) Alice wants to send M to Bob

ii) In public directory, Alice looks up Bob's Public Key, P_B

iii) Alice sends $P_B(M)$ to Bob

iv) Bob uses his Secret Key, S_B to decrypt $M = S_B(P_B(M))$



Public-Key Cryptosystems

- No agreement in advance on a secret code.
- Sender (*Alice*) and receiver (*Bob*) each have both a **public key** and a **secret key**.
- KP_A: Alice's public key; KS_A: Alice's secret key.
  KP_B: Bob's public key; KS_B: Bob's secret key.
- Public key available to anyone (including eavesdroppers). Secret key is kept by owner.
- Functions associated with KS_A, KP_A, KS_B,KP_B are S_A, P_A, S_B, and P_B. S_A and P_A are inverses; S_B and P_B are inverses; So, for any message M

$$M = S_A(P_A(M)) = P_A(S_A(M)),$$
$$M = S_B(P_B(M)) = P_B(S_B(M)).$$

An unsecure public key cryptosystem

- Messages are numbers in range 1 to 999.
- Bob's public key is function $P_B(M) = rev(1000 - M)$;
  $rev()$ reverses the digits of a number.
- To encrypt $M = 167$, Alice sends Bob
  $C = rev(1000 - 167) = rev(833) = 338$.
- In this case, $S_B(C) = 1000 - rev(C)$,
  so Bob can easily decode the message.
- Problem: this is *Not* secure, because *anyone* who knows **public key,** P_B**,** can figure out **secret key** S_B.

Challenge: In order for a public-key cryptosystem to work we must be able to find **public/secret key pairs** such that

- Receiver Bob can easily calculate $S_B(X)$
- No one else knowing **public key,** P_B**,** will easily be able to figure out our **secret key,** S_B.

Constructing such **pubic/secret key pairs** sounds almost impossible. Surprisingly, in the mid 1970s, Rivest, Shamir and Adelman, figured out how to do this using simple modular arithmetic.

The result is the **RSA Public Key Cryptosystem**, which is the basis for most e-commerce. We will learn its details in the lecture following the next one.