

# Discrete Mathematics for Computer Science

## Lecture 10: RAS Cryptosystem and Mathematical Induction

Dr. Ming Tang

Department of Computer Science and Engineering  
Southern University of Science and Technology (SUSTech)  
Email: tangm3@sustech.edu.cn



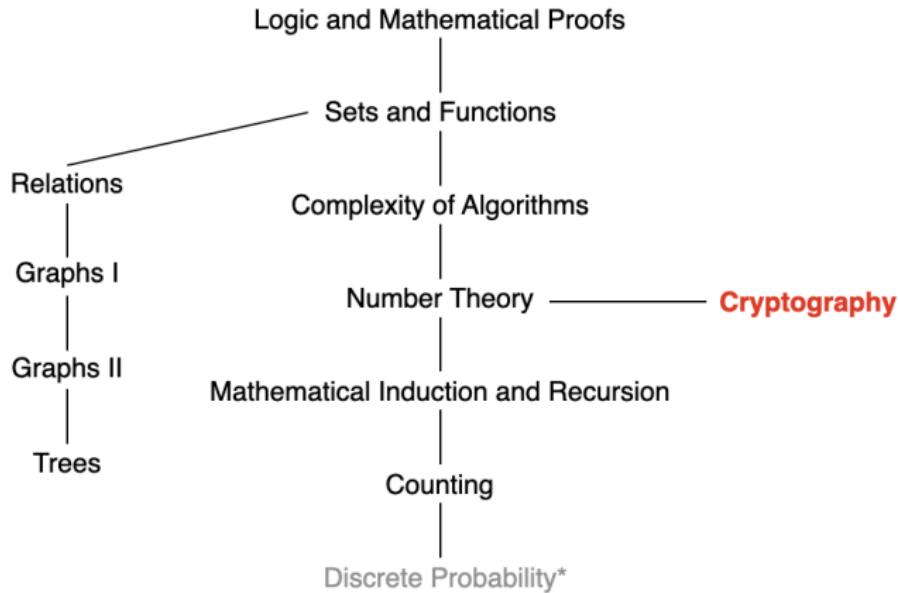
# Number of Solutions to Congruences

**Theorem:** Let  $\gcd(a, m) = d$ . Let  $m' = m/d$  and  $a' = a/d$ . The congruence  $ax \equiv b \pmod{m}$  has solutions if and only if  $d \mid b$ .

- If  $d \mid b$ , then there are exactly  $d$  “solutions”, where by “solution” we mean a congruence class mod  $m$ .
- If  $x_0$  is a solution, then the other solutions are given by  $x_0 + m', x_0 + 2m', \dots, x_0 + (d - 1)m'$ .

We will not cover this in the exam.

# This Lecture



Cryptography: classical cryptography, **RAS cryptosystem**



Southern University  
of Science and  
Technology

# Overview

- RSA as Public Key System
  - ▶ Only target recipient can decrypt the message:
  - ▶ Bob's public key and private key.



- RSA as Digital Signature
- Diffie-Hellman Key Exchange Protocol

# RAS Cryptosystem

Rivest-Shamir-Adleman

**2002 Turing Award**

2002

[Ronald L. Rivest](#),  
[Adi Shamir](#) and  
[Leonard M. Adleman](#)

For [their ingenious contribution](#) for making [public-key cryptography](#) useful in practice.

Pick two large primes  $p$  and  $q$ . Let  $n = pq$ . **Encryption key** ( $n, e$ ) and **decryption key** ( $n, d$ ) are selected such that

- $\gcd(e, (p-1)(q-1)) = 1$
- $ed \equiv 1 \pmod{(p-1)(q-1)}$

**RSA encryption:**  $C = M^e \pmod{n}$

**RSA decryption:**  $M = C^d \pmod{n}$

# RSA Encryption

- 1 Translate a plaintext message into integers, each with **two digits**, e.g., A is translated into 00, B into 01, . . . , and Z into 25.
- 2 Divide this string into **equally sized blocks** of  $2N$  digits
  - ▶  $2N$  is the largest even number such that the number 2525...25 with  $2N$  digits does not exceed  $n$ .
- 3 For each block, transform it into a ciphertext block:

$$C = M^e \bmod n$$

# RSA Encryption: Example

Encrypt the message “STOP” with key ( $n = 2537$ ,  $e = 13$ ). Note that  $2537 = 43 \cdot 59$ , where  $p = 43$  and  $q = 59$  are primes, and  $\gcd(e, (p - 1)(q - 1)) = 1$ .

## Solution:

- 1 Translate into integers: 18191415
- 2 Divide this into blocks of 4 digits (because  $2525 < 2537 < 252525$ ):  
1819 1415
- 3 Encrypt each block using the mapping

$$C = M^{13} \pmod{2537}.$$

We have  $1819^{13} \pmod{2537} = 2081$  and  $1415^{13} \pmod{2537} = 2182$ .  
The encrypted message is 2081 2182.

## RSA Decryption

For each block, transform the ciphertext into plaintext message:

$$M = C^d \pmod{n}$$

**Example:** What is the decrypted message of 0981 0461 with  $e = 13$ ,  $p = 43$ ,  $q = 59$ ?

**Solution:** Recall that  $ed \equiv 1 \pmod{(p-1)(q-1)}$ . Thus,  $d = 937$  is an inverse of 13 modulo  $42 \cdot 58 = 2436$ .

For each block, transform it into plaintext message:

$$M = C^{937} \pmod{2537}.$$

Since  $0981^{937} \pmod{2537} = 0704$  and  $0461^{937} \pmod{2537} = 1115$ , the plaintext message is 0704 1115, which is “HELP”.

# RAS Cryptosystem

Pick two large primes  $p$  and  $q$ . Let  $n = pq$ . **Encryption key** ( $n, e$ ) and **decryption key** ( $n, d$ ) are selected such that

- (1)  $\gcd(e, (p-1)(q-1)) = 1$
- (2)  $ed \equiv 1 \pmod{(p-1)(q-1)}$

**RSA encryption:**  $C = M^e \pmod{n}$ ;

**RSA decryption:**  $M = C^d \pmod{n}$ . Why?

According to (1), the inverse  $d$  exists. According to (2), there exists an integer  $k$  such that

$$de = 1 + k(p-1)(q-1).$$

It follows that  $C^d \equiv (M^e)^d = M^{de} = M^{1+k(p-1)(q-1)} \pmod{n}$ .

# RAS Cryptosystem

Assuming that  $\gcd(M, p) = \gcd(M, q) = 1$ , we have  $M^{p-1} \equiv 1 \pmod{p}$  and  $M^{q-1} \equiv 1 \pmod{q}$ . (see Theorem 3 in Section 4.4)

**FERMAT'S LITTLE THEOREM** If  $p$  is prime and  $a$  is an integer not divisible by  $p$ , then

$$a^{p-1} \equiv 1 \pmod{p}.$$

Furthermore, for every integer  $a$  we have

$$a^p \equiv a \pmod{p}.$$

## RAS Cryptosystem

According to (1), the inverse  $d$  exists. According to (2), there exists an integer  $k$  such that

$$de = 1 + k(p-1)(q-1).$$

It follows that  $C^d \equiv (M^e)^d = M^{de} = M^{1+k(p-1)(q-1)} \pmod{n}$ .

Assuming that  $\gcd(M, p) = \gcd(M, q) = 1$ , we have  $M^{p-1} \equiv 1 \pmod{p}$  and  $M^{q-1} \equiv 1 \pmod{q}$ .

$$C^d \equiv M \cdot (M^{p-1})^{k(q-1)} \equiv M \cdot 1 = M \pmod{p}$$

$$C^d \equiv M \cdot (M^{q-1})^{k(p-1)} \equiv M \cdot 1 = M \pmod{q}.$$

Because  $\gcd(p, q) = 1$ , we have

$$C^d \equiv M \pmod{pq}.$$

This basically implies that

$$M = C^d \pmod{n}$$

# RSA as Public Key System

Pick two large primes  $p$  and  $q$ . Let  $n = pq$ . **Encryption key** ( $n, e$ ) and **decryption key** ( $n, d$ ) are selected such that

- (1)  $\gcd(e, (p-1)(q-1)) = 1$
- (2)  $ed \equiv 1 \pmod{(p-1)(q-1)}$

**RSA encryption:**  $C = M^e \pmod{n}$ ;

**RSA decryption:**  $M = C^d \pmod{n}$ .

## RSA as a Public Key System

- Public key:  $(n, e)$
- Private key:  $d$
- $p, q$  must be kept **secret**!

Why is the RSA cryptosystem suitable for public key cryptography?

# RSA as Public Key System

## RSA as a Public Key System

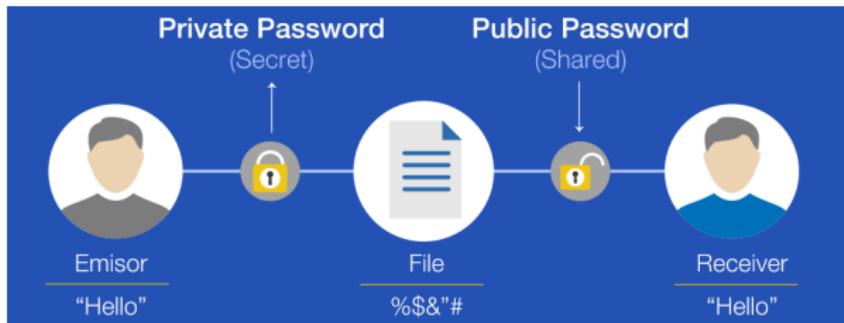
- Public key:  $(n, e)$ ; Private key:  $(n, d)$
- $p, q$  must be kept **secret**!

Why is the RSA cryptosystem suitable for public key cryptography?

- It is possible to **rapidly construct** a public key by finding two large primes  $p$  and  $q$ , each with more than 200 digits.
- When we know  $p$  and  $q$ , we can **quickly find** an inverse  $d$ .
- However, **no method** is known to decrypt messages that is not based on finding a factorization of  $n$ .
  - ▶ **Factorization** is believed to be a **difficult problem**.
  - ▶ The most efficient factorization methods known (as of 2010) require billions of years to factor 400-digit integers.

# Overview

- RSA as Public Key System
- RSA as Digital Signature
  - ▶ The recipient of the message knows that it came from the person they think it came from.



- Diffie-Hellman Key Exchange Protocol

# RSA as Digital Signature

Alice's RSA public key is  $(n, e)$  and her private key is  $d$ .

Alice splits the plaintext message into blocks and applies her decryption function:

$$S = M^d \bmod n \quad (\text{RSA signature})$$

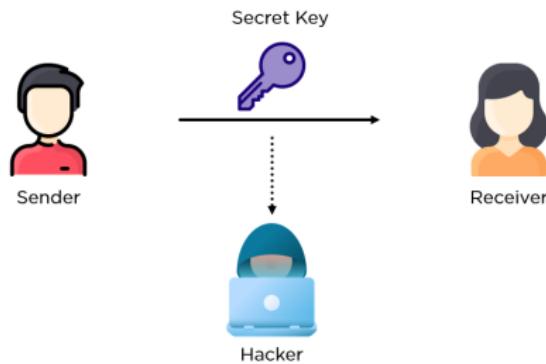
When a recipient receives her message, they apply Alice's encryption function:

$$M = S^e \bmod n \quad (\text{RSA verification})$$

Alice can send her message to as many people as she wants and by signing it in this way, **every recipient can be sure it came from Alice.**

# Overview

- RSA as a Public Key System
- RSA as Digital Signature
- Diffie-Hellman Key Exchange Protocol
  - ▶ Exchange a secret key over an insecure communications channel



# Diffie-Hellman Key Exchange Protocol

Two parties exchange a **secret key** over an **insecure** communications channel **without** having shared any information in the past.

## Diffie-Hellman Key Exchange Protocol

Before introducing the protocol:

**Definition:** A **primitive root modulo a prime  $p$**  is an integer  $r$  in  $\mathbb{Z}_p$  such that every nonzero element of  $\mathbb{Z}_p$  is a power of  $r$ .

**Example:** Whether 2 is a primitive root modulo 11?

When we compute the powers of 2 in  $\mathbb{Z}_{11}$ , we obtain  $2^1 = 2$ ,  $2^2 = 4$ ,  $2^3 = 8$ ,  $2^4 = 5$ ,  $2^5 = 10$ ,  $2^6 = 9$ ,  $2^7 = 7$ ,  $2^8 = 3$ ,  $2^9 = 6$ ,  $2^{10} = 1$ .

Because every element of  $\mathbb{Z}_{11}$  is a power of 2, 2 is a primitive root of 11.

# Diffie-Hellman Key Exchange Protocol

Suppose that Alice and Bob want to share a common key. Consider  $\mathbb{Z}_p$ .

- (1) Alice and Bob agree to use a prime  $p$  and a primitive root  $a$  of  $p$ .
- (2) Alice chooses a secret integer  $k_1$  and sends  $a^{k_1} \bmod p$  to Bob.
- (3) Bob chooses a secret integer  $k_2$  and sends  $a^{k_2} \bmod p$  to Alice.
- (4) Alice computes  $(a^{k_2})^{k_1} \bmod p$ .
- (5) Bob computes  $(a^{k_1})^{k_2} \bmod p$ .

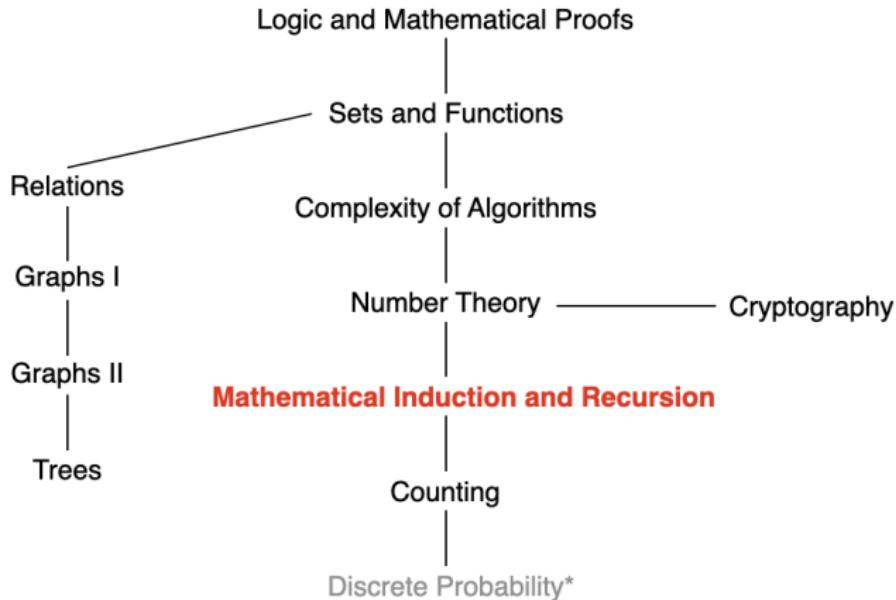
Alice and Bob have computed their shared key:

$$(a^{k_2})^{k_1} \bmod p = (a^{k_1})^{k_2} \bmod p.$$

- Public information:  $p$ ,  $a$ ,  $a^{k_1} \bmod p$ , and  $a^{k_2} \bmod p$
- Secret:  $k_1$ ,  $k_2$ ,  $(a^{k_2})^{k_1} \bmod p = (a^{k_1})^{k_2} \bmod p$

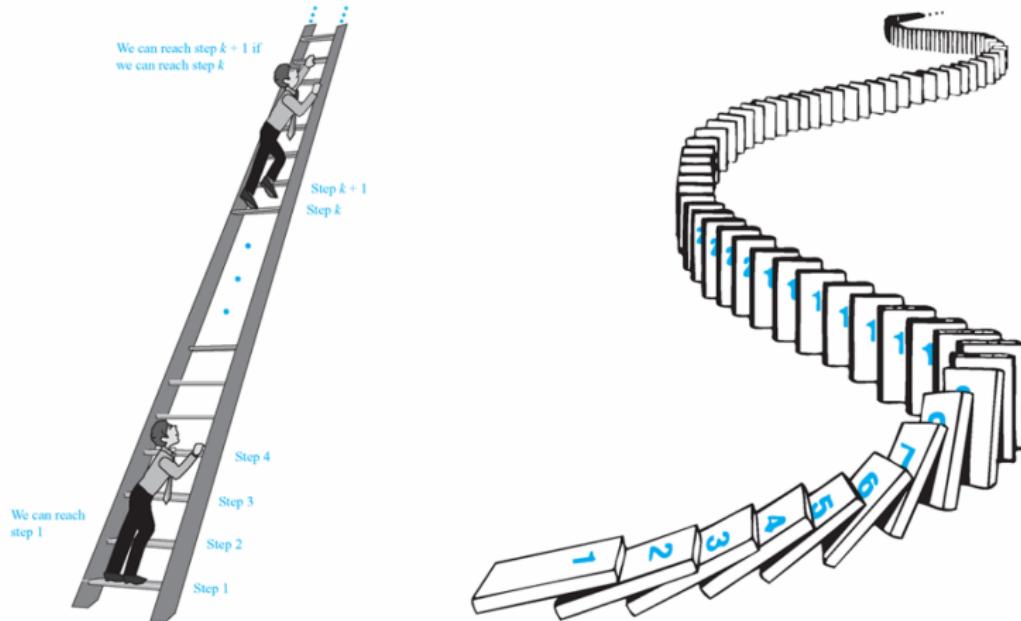
Note that it is very hard to determine  $k_1$  with  $a$ ,  $p$ , and  $a^{k_1} \bmod p$ .

# This Lecture



Mathematical induction, recursion, ....

# Mathematical Induction



# Mathematical Induction

We start by reviewing proof by **smallest counterexample** to try and understand what it is really doing.

This leads us to transform the **indirect proof** of proof by counterexample to **direct proof**. This direct proof technique will be **induction**.

We conclude by distinguishing between the **weak principle** of mathematical induction and the **strong principle** of mathematical induction.

- The strong principle can actually be derived from the weak principle.

# Mathematical Induction

The statement  $P(n)$  is true for all  $n = 0, 1, 2, \dots$

We prove this by

- (i) Assume that a counterexample exists, i.e., There is some  $n > 0$  for which  $P(n)$  is false.
- (ii) Let  $m > 0$  be the smallest value for which  $P(m)$  is false
- (iii) Then, use the fact that  $P(m')$  is true for all  $0 \leq m' < m$  to show that  $P(m)$  is true, **contradicting** the choice of  $m$ .

Contradiction!

## Example 1

Use proof by **smallest counterexample** to show that,  $\forall n \in N$ ,

$$0 + 1 + 2 + 3 + \dots + n = \frac{(n+1)n}{2}.$$

We use  $(*)$  to denote this equation.

- Suppose that  $(*)$  is **not** always true.
- Then there must be a smallest  $m \in N$  such that  $(*)$  does not hold for  $m$ .
- For any nonnegative integer  $n < m$ ,

$$0 + 1 + 2 + 3 + \dots + n = \frac{(n+1)n}{2}$$

Note that since the equation always holds for 0 (because

$0 = (0+1) \times 0/2$ ), The smallest counterexample  $m$  is **larger than 0**



Southern University  
of Science and  
Technology

## Example 1

We now have

- (i) smallest counterexample  $m$  is greater than 0, and
- (ii) (\*) holds for  $m - 1$

Substituting  $m - 1$  for  $n$  gives

$$0 + 1 + 2 + 3 + \dots + m - 1 = \frac{(m - 1)m}{2}$$

Adding  $m$  to both sides gives

$$0 + 1 + 2 + 3 + \dots + m - 1 + m = \frac{(m - 1)m}{2} + m = \frac{(m + 1)m}{2}.$$

Thus,  $m$  is not a counterexample. **Contradiction!**

Therefore, (\*) holds for all positive integers  $m$ .

## Example 1: Insights

The key step were

- $P(0)$  is true such that the smallest counterexample  $m$  exists
- proving that

$$P(m - 1) \rightarrow P(m)$$

Recall that  $P(n)$  is the statement

$$0 + 1 + 2 + 3 + \dots + n = \frac{(n + 1)n}{2}.$$

## Example 2

Use proof by **smallest counterexample** to show that,  $\forall n \in \mathbb{N}$ ,

$$2^{n+1} \geq n^2 + 2.$$

Let  $P(n)$  denote  $2^{n+1} \geq n^2 + 2$ .

We start by assuming that the statement  $\forall n P(n)$  is **false**.

When a **universal quantifier** is false, there must be some  $n$  for which it is false.

## Example 2

Let  $m$  be the **smallest** nonnegative integer for which  $2^{m+1} \not\geq m^2 + 2$ .

This means that, for all  $n \in N$  with  $n < m$ ,

$$2^{n+1} \geq n^2 + 2.$$

Since  $2^{0+1} \geq 0^2 + 2$ , we know that  $m > 0$ . Thus,  $m - 1$  is a nonnegative integer less than  $m$ .

Then setting  $n = m - 1$  gives

$$2^{(m-1)+1} \geq (m-1)^2 + 2$$

or

$$2^m \geq m^2 - 2m + 1 + 2 = m^2 - 2m + 3. \quad (*)$$

## Example 2

From (\*), we are now given  $2^m \geq m^2 - 2m + 3$ .

Multiply both sides by 2, giving

$$2^{m+1} = 2 \cdot 2^m \geq 2 \cdot (m^2 - 2m + 3) = 2m^2 - 4m + 6.$$

To get a contradiction, we want to convert the right side into  $m^2 + 2$  plus an additional nonnegative term.

Thus, we write

$$2^{m+1} \geq 2m^2 - 4m + 6 = m^2 + 2 + (m - 2)^2 \geq m^2 + 2$$

contradiction!

## Example 2

Let  $P(n)$  denote  $2^{n+1} \geq n^2 + 2$ . We just showed that

- (a)  $P(0)$  is true
- (b) If  $n > 0$ , then  $P(n - 1) \rightarrow P(n)$

What did we do?

- Suppose there is some  $n$  for which  $P(n)$  is false (\*)
- Let  $m$  be the smallest counterexample
- From (a)  $m > 0$ , so  $P(m - 1)$  is true
- From (b), using direct inference,  $P(m)$  is true
- This leads to contradiction.
- Thus,  $P(n)$  is true for all  $n \in N$ .

## Example 2

Let  $P(n)$  denote  $2^{n+1} \geq n^2 + 2$ . We just showed that

- (a)  $P(0)$  is true
- (b) If  $n > 0$ , then  $P(n - 1) \rightarrow P(n)$

**Main idea:** We used proof by smallest counterexample to derive that  $P(n)$  is true for all  $n \in N$ .

This is an **indirect** proof. Is it possible to prove this fact **directly**?

Since  $P(0)$  and  $P(n - 1) \rightarrow P(n)$ , we see that

$P(0)$  implies  $P(1)$ ,  $P(1)$  implies  $P(2)$ , ...

# The Principle of Mathematical Induction

**Well-Ordering Property:** Every nonempty set of nonnegative integers has a least element.

The well-ordering property permits us to assume that every set of nonnegative integers has a smallest element, allowing us to use the smallest counterexample.

This is actually equivalent to the principle of mathematical induction.

## Principle. (Weak Principle of Mathematical Induction)

- (a) **Basic Step:** the statement  $P(b)$  is true
- (b) **Inductive Step:** the statement  $P(n - 1) \rightarrow P(n)$  is true for all  $n > b$

Thus,  $P(n)$  is true for all integers  $n \geq b$ .

## Proof by Induction: Example 1

For all  $n \geq 0$ ,  $2^{n+1} \geq n^2 + 2$

Let  $P(n)$  denote  $2^{n+1} \geq n^2 + 2$ .

- (i) Note that for  $n = 0$ ,  $2^{0+1} = 2 \geq 2 = 0^2 + 2$ , which is  $P(0)$
- (ii) Suppose that  $n > 0$  and that  $2^{(n-1)+1} \geq (n-1)^2 + 2$  (\*)

$$\begin{aligned} 2^{n+1} &\geq 2(n-1)^2 + 4 \\ &= (n^2 + 2) + (n^2 - 4n + 4) \\ &= n^2 + 2 + (n-2)^2 \\ &\geq n^2 + 2 \end{aligned}$$

Hence, we have just proven that for  $n > 0$ ,  $P(n-1) \rightarrow P(n)$ .

By mathematical induction,  $\forall n \geq 0, 2^{n+1} \geq n^2 + 2$ .

## Proof by Induction: Example 2

For all  $n \geq 2$ ,  $2^{n+1} \geq n^2 + 3$

Let  $P(n)$  denote  $2^{n+1} \geq n^2 + 3$ .

- (i) **Basis Step:** Note that for  $n = 2$ ,  $2^{2+1} = 8 \geq 7 = 2^2 + 3$ , which is  $P(2)$
- (ii) **Inductive Step:** Suppose that  $n > 2$  and that  $2^{n-1+1} \geq (n-1)^2 + 3$   
(\*) **(Inductive Hypothesis)**

$$\begin{aligned} 2^{n+1} &\geq 2(n-1)^2 + 6 \\ &= n^2 + 3 + n^2 - 4n + 4 + 1 \\ &= n^2 + 3 + (n-2)^2 + 1 \\ &> n^2 + 3 \end{aligned}$$

Hence, we have just proven that for  $n > 2$ ,  $P(n-1) \rightarrow P(n)$ .

By mathematical induction,  $\forall n > 2$ ,  $2^{n+1} \geq n^2 + 3$ . **(Inductive Conclusion)**

# Another Form of Induction

We may have **another form** of direct proof as follows.

- First suppose that we have proof of  $P(0)$
- Next suppose that we have a proof that,  $\forall n > 0$ ,

$$P(0) \wedge P(1) \wedge P(2) \wedge \dots \wedge P(n-1) \rightarrow P(n)$$

Then,

- Then,  $P(0)$  implies  $P(1)$
- $P(0) \wedge P(1)$  implies  $P(2)$
- $P(0) \wedge P(1) \wedge P(2)$  implies  $P(3) \dots$

Iterating gives us a proof of  $P(n)$  for all  $n$ .

# Strong Induction

**Principle** (Strong Principle of Mathematical Induction):

- (a) **Basic Step**: the statement  $P(b)$  is true
- (b) **Inductive Step**: for all  $n > b$ , the statement

$$P(b) \wedge P(b+1) \wedge \dots \wedge P(n-1) \rightarrow P(n) \text{ is true.}$$

Then,  $P(n)$  is true for all integers  $n \geq b$ .

# Mathematical Induction

In practice, we **do not** usually explicitly distinguish between the weak and strong forms.

In reality, they are **equivalent to each other** in that the weak form is a special case of the strong form, and the strong form can be derived from the weak form.

# Summary

A **typical** proof by mathematical induction, showing that a statement  $P(n)$  is true for all integers  $n \geq b$  consists of three steps:

- **Base Step:** We show that  $P(b)$  is true.
- **Inductive Step:** We then,  $\forall n > b$ , show either of the following

$$P(n - 1) \rightarrow P(n)$$

$$P(b) \wedge P(b + 1) \wedge \dots \wedge P(n - 1) \rightarrow P(n)$$

We need to make the inductive hypothesis of either  $P(n - 1)$  or  $P(b) \wedge P(b + 1) \wedge \dots \wedge P(n - 1)$ . We then derive  $P(n)$ .

- We conclude on the basis of the principle of mathematical induction that  $P(n)$  is true for all  $n \geq b$ .

## Generalization: to be discussed in later sections

A Relation  $a_0 \prec a_1 \prec \dots$  with a least element:

A **typical** proof by mathematical induction, showing that a statement  $P(a_n)$  is true for all integers  $n \geq 0$  consists of three steps:

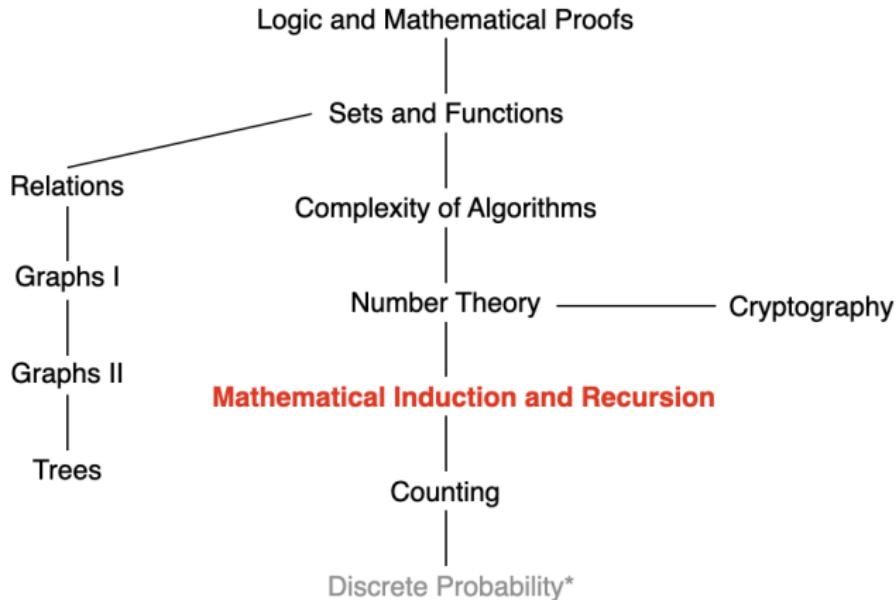
- **Base Step:** We show that  $P(a_0)$  is true.
- **Inductive Step:** We then,  $\forall n > 0$ , show either of the following

$$P(a_{n-1}) \rightarrow P(a_n)$$

$$P(a_0) \wedge P(a_1) \wedge \dots \wedge P(a_{n-1}) \rightarrow P(a_n)$$

- We conclude on the basis of the principle of mathematical induction that  $P(a_n)$  is true for all  $n \geq 0$ .

# This Lecture



Mathematical induction, **recursion**, ....

# Overview

To design an algorithm or solve a problem

- using a recursive idea
- then, prove using mathematical induction

# Overview

- An Example: [Towers of Hanoi](#)
- Recurrence
- First-Order Linear Recurrence
- Growth Rates of Solutions to Recurrence

# Recursion

Recursive computer programs or algorithms often lead to inductive analysis.

- Solve a problem using a recursive algorithm
- Prove the correctness of the algorithm
- Analyze the algorithm in a recursive fashion → recursive equation
- Closed-form solution

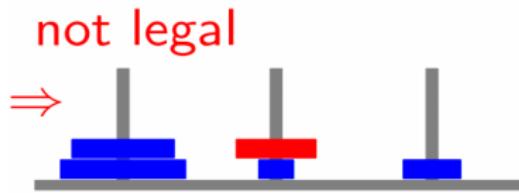
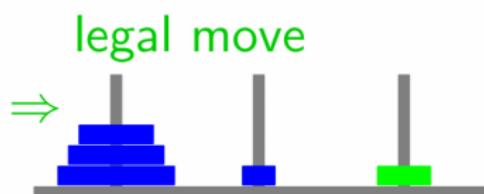
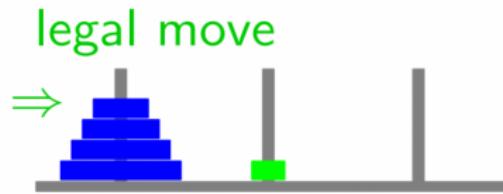
A classical example of recursion is the [Towers of Hanoi Problem](#).

# Towers of Hanoi



- 3 pegs,  $n$  disks of different sizes
- A **legal move** takes a disk from one peg and moves it onto another peg so that it is not on top of a smaller disk
- **Problem:** Find a way to move all of the disks from one peg to another

# Towers of Hanoi

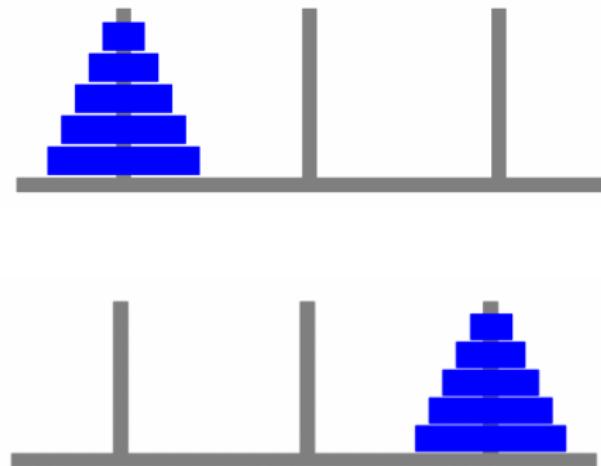


# Towers of Hanoi

**Problem:** Start with  $n$  disks on leftmost peg

using only legal moves

move all disks to rightmost peg.



Find an algorithm?

# Towers of Hanoi

Recursion Base:

If  $n = 1$ , moving one disk from  $i$  to  $j$  is easy. Just move it.

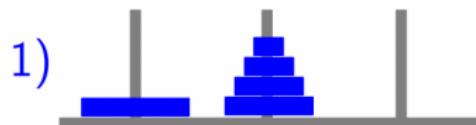


# Towers of Hanoi

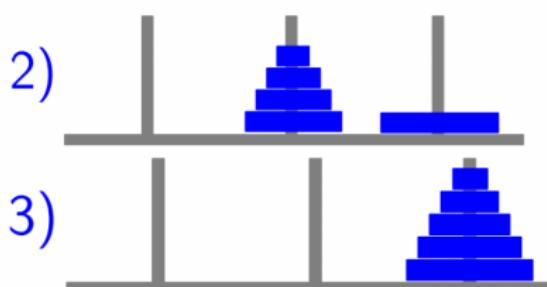
Given  $i, j \in \{1, 2, 3\}$ . Let  $\overline{\{i,j\}} = \{1, 2, 3\} - \{i\} - \{j\}$ . For example,  $\overline{\{1,2\}} = \{3\}$ .



To move  $n > 1$  disks from  $i$  to  $j$



move top  $n - 1$  disks from  $i$  to  $\overline{\{i,j\}}$



move largest disk from  $i$  to  $j$

move top  $n - 1$  disks from  $\overline{\{i,j\}}$  to  $j$



Southern University  
of Science and  
Technology

# Towers of Hanoi



```
3 public class Hanoi
4 {
5
6     public void move(int n, char a, char b, char c)
7     {
8         if (n == 1)
9             System.out.println("plate " + n + " from " + a + " to " + c);
10        else
11        {
12            move(n-1,a,c,b);
13            System.out.println("plate " + n + " from " + a + " to " + c);
14            move(n-1,b,a,c);
15        }
16    }
17 }
18 }
```

# Towers of Hanoi

To prove correctness of solution, we are implicitly using induction:

$p(n)$  is statement that algorithm is correct for  $n$

- $p(1)$  is statement that algorithm works for  $n = 1$  disks, which is obviously true
- $p(n - 1) \rightarrow p(n)$  is recursion statement that  
if our algorithm works for  $n - 1$  disks, then we can build a correct solution for  $n$  disks.

# Towers of Hanoi



**Running Time:**  $M(n)$  is number of disk moves needed for  $n$  disks.

- $M(1) = 1$
- if  $n > 1$ , then  $M(n) = 2M(n - 1) + 1$

Iterating the recurrence gives  $M(1) = 1$ ,  $M(2) = 3$ ,  $M(3) = 7$ ,  $M(4) = 15$ ,  $M(5) = 31$ , . . .

We guess that  $M(n) = 2^n - 1$ .

We'll prove this by [induction](#). Later, we'll also see how to solve without guessing.

# Towers of Hanoi

Formally, given

$$M(n) = \begin{cases} 1, & \text{if } n = 1 \\ 2M(n - 1) + 1, & \text{otherwise} \end{cases}$$

We show that  $M(n) = 2^n - 1$ .

**Proof.** (by induction)

The base case  $n = 1$  is true, since  $2^1 - 1 = 1$ .

For the inductive step, assume that  $M(n - 1) = 2^{n-1} - 1$  for  $n > 1$ .

Then  $M(n) = 2M(n - 1) + 1 = 2(2^{n-1} - 1) + 1 = 2^n - 1$

# Towers of Hanoi

Note that we used induction twice.

- The first time was to derive correctness of algorithm and the recurrence

$$M(n) = \begin{cases} 1, & \text{if } n = 1 \\ 2M(n - 1) + 1, & \text{otherwise} \end{cases} \quad (1)$$

- The second time was to derive the closed form solution  $M(n) = 2^n - 1$  of the recurrence.

# Overview

- Towers of Hanoi
- Recurrence
- First-Order Linear Recurrence
- Growth Rates of Solutions to Recurrence

# Recurrence

A **recurrence equation** or recurrence for a function:

- defined on the set of integers  $\geq b$
- tell us how to compute the  $n$ -th value from some or all the **first  $n - 1$  values**

To completely specify a function on the basis of a recurrence:

- **Basis step (initial condition):** Specify the value of the function at zero.
- **Recursive step:** Give a rule for finding its value at an integer from its values at smaller integers.

**Example:**  $F(0) = 1$ ,  $F(1) = 1$ ,  $F(n) = F_{n-1} + F_{n-2}$  for  $n \geq 2$ .

## Recurrences: Example

**Example 1:** Let  $S(n)$  be the number of subsets of a set of size  $n$ , i.e.,  $|\mathcal{P}(n)|$ . What is the formula for  $S(n)$ ?

Consider a set  $\mathcal{S}_n$  of size  $n$  and an arbitrary subset  $\mathcal{S}'_n \subseteq \mathcal{S}_n$ . Any element  $s \in \mathcal{S}_n$  can be either in set  $\mathcal{S}'_n$  or not in  $\mathcal{S}'_n$ .

Thus,  $S(n) = |\mathcal{P}(n)| = 2^n$ .

**How to think it recursively?**

## Recurrences: Example

Consider the subsets of  $\{1, 2\}$ :

$$\emptyset, \{1\}, \{2\}, \{1, 2\}$$

Consider the eight subsets of  $\{1, 2, 3\}$ :

$$\emptyset, \{1\}, \{2\}, \{1, 2\}, \{3\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}$$

It can be seen that

$$\begin{array}{cccc} \emptyset & \{1\} & \{2\} & \{1, 2\} \\ \{3\} & \{1, 3\} & \{2, 3\} & \{1, 2, 3\} \end{array}$$

This suggests that the recurrence for the number of subsets of an  $n$ -element set  $\{1, 2, \dots, n\}$  is  $S(0) = 1$  and  $S(n) = 2S(n - 1)$  for  $n \geq 1$ .

Closed-form?

# Iterating a Recurrence

Let  $T(n) = rT(n - 1) + a$ , where  $r$  and  $a$  are constants.

**Can we generalize this to find a closed-form solution?**

$$\begin{aligned} T(n) &= rT(n - 1) + a \\ &= r(rT(n - 2) + a) + a \\ &= r^2T(n - 2) + ra + a \\ &= r^2(rT(n - 3) + a) + ra + a \\ &= r^3T(n - 3) + r^2a + ra + a \\ &= r^3(rT(n - 4) + a) + r^2a + ra + a \\ &= r^4T(n - 4) + r^3a + r^2a + ra + a. \end{aligned}$$

**Guess**  $T(n) = r^n T(0) + a \sum_{i=0}^{n-1} r^i$ .

## Iterating a Recurrence

The method we used to guess the solution is called **iterating the recurrence**, because we **repeatedly** (iteratively) use the recurrence.

Another approach is to iterate from the “**bottom-up**” instead of “**top-down**”.

$$T(0) = b$$

$$T(1) = rT(0) + a = rb + a$$

$$T(2) = rT(1) + a = r(rb + a) + a = r^2b + ra + a$$

$$T(3) = rT(2) + a = r^3b + r^2a + ra + a$$

This would lead to the same **guess**:

$$T(n) = r^n b + a \sum_{i=0}^{n-1} r^i.$$

Summation for geometric sequence.

# Formula of Recurrences

**Theorem:** If  $T(n) = rT(n - 1) + a$ ,  $T(0) = b$ , and  $r \neq 1$ , then

$$T(n) = r^n b + a \frac{1 - r^n}{1 - r}$$

for all nonnegative integers  $n$ .

## How to prove?

Mathematical Induction:

- **Basis step:** We verify that  $T(0)$  holds .
- **Inductive step:** We show that the conditional statement “if  $T(n - 1)$  holds, then  $T(n)$  holds” for all  $n \geq 1$ .

## Formula of Recurrences

**Theorem:** If  $T(n) = rT(n - 1) + a$ ,  $T(0) = b$ , and  $r \neq 1$ , then

$$T(n) = r^n b + a \frac{1 - r^n}{1 - r}$$

for all nonnegative integers  $n$ .

- **Basis step:** We verify that  $T(0)$  holds:

$$T(0) = r^0 b + a \frac{1 - r^0}{1 - r}$$

- **Inductive step:** We show that the conditional statement “if  $T(n - 1)$  holds, then  $T(n)$  holds” for all  $n \geq 1$ :

Now assume that  $n > 0$  and

$$T(n - 1) = r^{n-1} b + a \frac{1 - r^{n-1}}{1 - r}.$$



Southern University  
of Science and  
Technology

# Formula of Recurrences

- **Basis step:** We verify that  $T(0)$  holds:
- **Inductive step:** We show that the conditional statement “if  $T(n - 1)$  holds, then  $T(n)$  holds” for all  $n \geq 1$ :

Now assume that  $n > 0$  and

$$T(n - 1) = r^{n-1}b + a\frac{1 - r^{n-1}}{1 - r}.$$

Thus,

$$\begin{aligned} T(n) &= rT(n - 1) + a \\ &= r\left(r^{n-1}b + a\frac{1 - r^{n-1}}{1 - r}\right) + a \\ &= r^n b + \frac{ar - ar^n}{1 - r} + a \\ &= r^n b + \frac{ar - ar^n + a - ar}{1 - r} \\ &= r^n b + a\frac{1 - r^n}{1 - r}. \end{aligned}$$

# Summary

An arbitrary problem, e.g., let  $S(n)$  be the number of subsets of a set of size  $n$ , i.e.,  $|\mathcal{P}(n)|$ . What is the formula for  $S(n)$ ?

- Consider the problem in a recursive fashion;

$$\begin{array}{cccc} \emptyset & \{1\} & \{2\} & \{1, 2\} \\ \{3\} & \{1, 3\} & \{2, 3\} & \{1, 2, 3\} \end{array}$$

- Recurrence equation: e.g.,  $S(0) = 1$ ,  $S(n) = 2S(n - 1)$  for  $n \geq 1$ ;
- Guess the closed-form;
- Use mathematical induction to prove.

# Overview

- Towers of Hanoi
- Recurrences
- First-Order Linear Recurrences
- Growth Rates of Solutions to Recurrences

# First-Order Linear Recurrences

A recurrence of the form  $T(n) = f(n)T(n - 1) + g(n)$  is called a **first-order linear recurrence**.

- **First Order:** because it only depends upon going back one step, i.e.,  $T(n - 1)$
- If it depends upon  $T(n - 2)$ , then it would be a **second-order recurrence**, e.g.,  $T(n) = T(n - 1) + 2T(n - 2)$ .
- **Linear:** because  $T(n - 1)$  only appears to the first power.
- Something like  $T(n) = (T(n - 1))^2 + 3$  would be a **non-linear first-order recurrence relation**.

# First-Order Linear Recurrences

$$T(n) = f(n)T(n - 1) + g(n)$$

When  $f(n)$  is a constant, say  $r$ , the general solution is almost as easy as we derived before. Iterating the recurrence gives

$$\begin{aligned} T(n) &= rT(n - 1) + g(n) \\ &= r(rT(n - 2) + g(n - 1)) + g(n) \\ &= r^2T(n - 2) + rg(n - 1) + g(n) \\ &= r^3T(n - 3) + r^2g(n - 2) + rg(n - 1) + g(n) \\ &\vdots \\ &= r^nT(0) + \sum_{i=0}^{n-1} r^i g(n - i) \end{aligned}$$

# First-Order Linear Recurrences

**Theorem:** For any positive constants  $a$  and  $r$ , and any function  $g$  defined on nonnegative integers, the solution to the first-order linear recurrence

$$T(n) = \begin{cases} rT(n-1) + g(n), & \text{if } n > 0 \\ a, & \text{if } n = 0 \end{cases}$$

is

$$T(n) = r^n a + \sum_{i=1}^n r^{n-i} g(i).$$

**Proof by induction**

## Example 1

Solve  $T(n) = 4T(n - 1) + 2^n$  with  $T(0) = 6$ .

$$\begin{aligned} T(n) &= 6 \cdot 4^n + \sum_{i=1}^n 4^{n-i} \cdot 2^i \\ &= 6 \cdot 4^n + 4^n \sum_{i=1}^n 4^{-i} \cdot 2^i \\ &= 6 \cdot 4^n + 4^n \sum_{i=1}^n \left(\frac{1}{2}\right)^i \\ &= 6 \cdot 4^n + \left(1 - \frac{1}{2^n}\right) \cdot 4^n \\ &= 7 \cdot 4^n - 2^n. \end{aligned}$$

## Example 2

Solve  $T(n) = 3T(n - 1) + n$  with  $T(0) = 10$ .

$$\begin{aligned} T(n) &= 10 \cdot 3^n + \sum_{i=1}^n 3^{n-i} \cdot i \\ &= 10 \cdot 3^n + 3^n \sum_{i=1}^n i \cdot 3^{-i} \end{aligned}$$

**Theorem.** For any real number  $x \neq 1$ ,

$$\sum_{i=1}^n ix^i = \frac{nx^{n+2} - (n+1)x^{n+1} + x}{(1-x)^2}.$$

## Example 2

Solve  $T(n) = 3T(n - 1) + n$  with  $T(0) = 10$ .

$$\begin{aligned} T(n) &= 10 \cdot 3^n + \sum_{i=1}^n 3^{n-i} \cdot i \\ &= 10 \cdot 3^n + 3^n \sum_{i=1}^n i \cdot 3^{-i} \\ &= 10 \cdot 3^n + 3^n \left( -\frac{3}{2}(n+1)3^{-(n+1)} - \frac{3}{4}3^{-(n+1)} + \frac{3}{4} \right) \\ &= \frac{43}{4}3^n - \frac{n+1}{2} - \frac{1}{4}. \end{aligned}$$

# Overview

- Towers of Hanoi
- Recurrences
- First-Order Linear Recurrences
- Growth Rates of Solutions to Recurrences

# Growth Rates of Solutions to Recurrences

- Divide and conquer algorithms
- Iterating recurrences
- Three different behaviors

# Divide and conquer algorithms

We just analyzed recurrences of the form

$$T(n) = \begin{cases} b, & \text{if } n = 0 \\ rT(n - 1) + a, & \text{if } n > 0 \end{cases}$$

These corresponded to the analysis of recursive algorithms in which a problem of size  $n$  is solved by recursively solving a problem of size  $n - 1$ .

We will now look at recurrences of the form

$$T(n) = \begin{cases} \text{something given,} & \text{if } n \leq n_0 \\ rT(n/m) + a, & \text{if } n > n_0 \end{cases}$$

## Example: Binary Search

Someone has chosen a number  $x$  between 1 and  $n$ . We need to discover  $x$ .

We are only allowed to ask two types of questions:

- Is  $x$  equal to  $k$ ?
- Is  $x$  greater than  $k$ ?

Our strategy will be to always ask **greater than** questions, at each step halving our search range, until the range only contains one number, when we ask a **final equal** to question.

## Example: Binary Search

---

1                    32                    48                    64

---

1                    32                    48                    64

Is  $x > 32$ ?

## Example: Binary Search

### Why this is a good approach?

**Method:** Each guess reduces the problem to one in which the range is only half as big.

This divides the original problem into one that is only half as big; we can now (recursively) conquer this smaller problem.

**Note:** When  $n$  is a power of 2, the number of questions in a binary search on  $[1, n]$ , satisfies

$$T(n) = \begin{cases} 1, & \text{if } n = 1 \\ T(n/2) + 1, & \text{if } n \geq 2 \end{cases}$$

This can also be proven inductively.

## Binary Search Example

$T(n)$ : number of questions in a binary search on  $[1, n]$

Assume:  $n$  is a power of 2. Give recurrence for  $T(n)$

$$T(n) = \begin{cases} 1, & \text{if } n = 1 \\ T(n/2) + 1, & \text{if } n \geq 2 \end{cases}$$

The number of questions needed for binary search on  $n$  items is:

- first step
- time to perform binary search on the remaining  $n = 2$  items

To compute the number of questions (suppose  $n$  is the power of 2)

- Basis step:  $T(1) = 1$  to ask: “Is the number  $k$ ?”
- Inductive step:  $T(n) = T(n/2) + 1$

# Growth Rates of Solutions to Recurrences

- Divide and conquer algorithms
- Iterating recurrences: how to solve

$$T(n) = \begin{cases} \text{something given,} & \text{if } n \leq n_0 \\ rT(n/m) + a, & \text{if } n > n_0 \end{cases}$$

- Three different behaviors