归纳与递归

回顾

问题1：什么是(初等)数论?

- 研究整数的性质：整除、余数、同余算术

问题2：素数有哪些性质?

- 素数基本定理
- 最大公约数
- 同余方程
- 欧拉定理/费马小定理

本节提要

- 归纳：
  - 数学归纳法与强数学归纳法
  - 运用良序公理来证明
- 递归：
  - 递归定义
  - 结构归纳法与递归算法

数学归纳法

- 证明目标
  - $\forall n\ P(n)$　//n的论域为正整数集合
- 证明框架
  - 基础步骤：$P(1)$为真
  - 归纳步骤：证明$\forall k\ (P(k) \rightarrow P(k+1))$
    - //对任意正整数k，给出 $P(k) \vdash P(k+1)$ 的论证步骤.
    - …
  - 因此，对任意正整数n, $P(n)$ 成立. // 即：$\forall n\ P(n)$

数学归纳法（有效性）

- 良序公理
  - 正整数集合的非空子集都有一个最小元素
- 数学归纳法的有效性（归谬法）
  - 假设$\forall n\, P(n)$不成立，则 $\exists n\, (\neg P(n))$成立.
  - 令S=$\{ n \in Z^+ \mid \neg P(n)\}$，S是非空子集.
  - 根据良序公理，S有最小元素，记为m, $m \neq 1$
  - $(m-1) \notin S$, 即$P(m-1)$成立.
  - 根据归纳步骤，$P(m)$成立，即$m \notin S$，矛盾.
  - 因此，$\forall n\, P(n)$成立.

数学归纳法（举例）

- $H_k=1+1/2+\ldots+1/k$ （k为正整数）
- 证明：$H_2^n \geq 1+n/2$ （n为正整数）
  - 基础步骤：$P(1)$为真，$H_2=1+1/2$
  - 归纳步骤：对任意正整数k, $P(k) \Rightarrow P(k+1)$.

    $H_2^{k+1} = H_2^k + 1/(2^k+1)+\ldots+1/2^{k+1}$

    $\geq (1+k/2)+2^k(1/2^{k+1}) = 1+(1+k)/2$
  - **因此，**对任意正整数n, $P(n)$ 成立.

数学归纳法（举例）

- 猜测前n个奇数的求和公式，并证明之。
  - 1=1
  - 1+3=4
  - 1+3+5=9
  - 1+3+5+7=16
  - …
  - $1+3+\ldots+(2n-1)=n^2$（n为正整数）

运用数学归纳法时犯的错误

- 平面上任何一组相互间不平行的直线必相交于一点.
  - 基础步骤：$P(2)$为真
  - 归纳步骤：对任意正整数k, $P(k) \vdash P(k+1)$.
    - 前k条交于p_1.
    - 后k条交于p_2.
    - $p_1 = p_2$

强数学归纳法

- 证明目标
  - $\forall n\ P(n)$　//n的论域为正整数集合
- 证明框架
  - 基础步骤：$P(1)$为真
  - 归纳步骤：证明$\forall k\,(P(1)\wedge\ldots\wedge P(k)\rightarrow P(k+1))$
    - //对任意正整数k，给出 $P(1), \ldots, P(k) \vdash P(k+1)$ 的论证步骤.
    - …
  - 因此，对任意正整数n, $P(n)$ 成立. // 即：$\forall n\ P(n)$

强数学归纳法（一般形式）

- 设$P(n)$是与整数n有关的陈述，a和b是两个给定的整数，且$a \le b$.
- 如果能够证明下列陈述
  - $P(a), P(a+1), \ldots, P(b)$.
  - 对任意$k \ge b$, $P(a) \wedge \ldots \wedge P(k) \rightarrow P(k+1)$
- 则下列陈述成立
  - 对任意$n \ge a$, $P(n)$.

强数学归纳法（有效性）

- $\{n \in Z \mid n \geq a\}$是良序的
  - 良序集：该集合的非空子集都有一个最小元素
- 数学归纳法的有效性（归谬法）
  - 假设$\forall n\, P(n)$不成立，则 $\exists n\, (\neg P(n))$成立.
  - 令$S=\{ n \in Z \mid (n \geq a) \wedge \neg P(n) \}$，S是非空子集.
  - 根据良序公理，S有最小元素，记为m, $m>a$
  - $a, \ldots, (m-1) \notin S$, 即$P(a), \ldots, P(m-1)$成立.
  - 根据归纳步骤，$P(m)$成立，即$m \notin S$，矛盾.
  - **因此，**$\forall n\, P(n)$成立.

强数学归纳法（举例）

- 任意整数**n(n ≥2)**可分解为（若干个）素数的乘积
  - **n = 2.**
  - 考察 **n+1.**
- 用**4**分和**5**分就可以组成**12**分及以上的每种邮资.
  - ***P*(12), *P*(13), *P*(14), *P*(15).**
  - 对任意***k*** **≥15,** ***P*(12)∧… ∧*P*(*k*)→*P*(*k*+1)**

Odd Pie Fights





- **Placing an odd number of people in the plane, in such a way that every pair of people has a distinct distance between them. At a signal, each person will throw a pie at the closest other person.**
- **At least one person does not get hit with a pie?**

二项式定理

Let $n \in \mathbb{N}$ and $x \in \mathbb{R}$. We have

$$(1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k.$$

We use induction on n. That is, we let

$$P(n) : \forall x \in \mathbb{R},\ (1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k.$$

We shall prove $P(1)$ is true, and prove

$$\forall k \in \mathbb{N},\ P(k) \implies P(k+1).$$

Again, proof for $P(1)$ is trivial, as $P(1)$ states

$$\forall x \in \mathbb{R},\ 1+x = \binom{1}{0} x^0 + \binom{1}{1} x^1$$

Next, let $k \in \mathbb{N}$ and assume $P(k)$. We prove $P(k+1)$. Let $x \in \mathbb{R}$. We have

$$\begin{aligned}(1+x)^{k+1} &= (1+x)^k (1+x) \\ \text{[by I.H.]} &= \left(\sum_{j=0}^{k} \binom{k}{j} x^j \right) (1+x) \\ &= \sum_{j=0}^{k} \binom{k}{j} x^j + \sum_{j=0}^{k} \binom{k}{j} x^{j+1}.\end{aligned}$$

We are going to merge these two sums.

二项式定理

To get an idea, let us consider an explicit example when $k = 2$:

$$\sum_{j=0}^{2} \binom{2}{j} x^j + \sum_{j=0}^{2} \binom{2}{j} x^{j+1}$$
$$= \left[\binom{2}{0} + \binom{2}{1}x + \binom{2}{2}x^2\right] + \left[\binom{2}{0}x + \binom{2}{1}x^2 + \binom{2}{2}x^3\right]$$
$$= \binom{2}{0} + \left[\binom{2}{1} + \binom{2}{0}\right]x + \left[\binom{2}{2} + \binom{2}{1}\right]x^2 + \binom{2}{2}x^3.$$

In general, we have

$$\sum_{j=0}^{k} \binom{k}{j} x^j + \sum_{j=0}^{k} \binom{k}{j} x^{j+1}$$
$$= 1 + \sum_{j=1}^{k} \binom{k}{j} x^j + \sum_{j=1}^{k} \binom{k}{j-1} x^j + x^{k+1}$$
$$= 1 + \sum_{j=1}^{k} \left[\binom{k}{j} + \binom{k}{j-1}\right] x^j + x^{k+1}.$$

Keep in mind that we want to prove this is equal to $\sum_{j=0}^{k+1} \binom{k+1}{j} x^j$, which is

$$1 + \sum_{j=1}^{k} \binom{k+1}{j} x^j + x^{k+1}.$$

Therefore, to finish the proof, it suffices to show that

$$\binom{k}{j} + \binom{k}{j-1} = \binom{k+1}{j}$$

$$\binom{k}{j} + \binom{k}{j-1} = \frac{k!}{j!(k-j)!} + \frac{k!}{(j-1)!(k-j+1)!}$$
$$= \frac{k!}{(j-1)!(k-j)!} \cdot \left[\frac{1}{j} + \frac{1}{k-j+1}\right]$$
$$= \frac{k!}{(j-1)!(k-j)!} \cdot \frac{k+1}{j(k-j+1)}$$
$$= \binom{k+1}{j}.$$

运用良序公理来证明（举例）

- 设a是整数, d是正整数, 则存在唯一的整数q和r满足
  - $0 \le r < d$
  - $a = dq + r$
- 证明
  - 令$S=\{a-dq \mid 0 \le a-dq\ ,\ q \in Z\}$，S非空.
  - 非负整数集合具有良序性
  - S有最小元，记为$r_0 = a-dq_0$.
  - 可证 $0 \le r_0 < d$
  - 唯一性证明， $0 \le r_1 - r_0 = d(q_0-q_1) < d$，因此，$q_1=q_0$

运用良序公理来证明（举例）

- 在循环赛胜果图中，若存在长度为m（$m\geq 3$）的回路，则必定存在长度为3的回路。

  备注：$a_i \rightarrow a_j$表示a_i赢了a_j

证明

- 设最短回路的长度为k //良序公理的保证
- 假设 $k>3$
- $a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow \ldots \rightarrow a_k \rightarrow a_1$
- 若$a_3 \rightarrow a_1$, 存在长度为3的回路，矛盾。
- 若$a_1 \rightarrow a_3$, 存在长度为$(k-1)$的回路，矛盾。

良序定理

- 我们是否也能够在任意集合上都建立良序呢？这个问题是如此重要，以致于希尔伯特在第一次国际数学大会上把它列为解决希尔伯特第一问题的先决条件。

- 让人惊讶的是，在此之后不久，策梅洛便解决了这个问题。更精确地说，策梅洛使用选择公理证明了所谓良序定理。今天，我们知道：选择公理、良序定理，还有一个著名的引理称为“左恩引理”，三者是互相等价的。我们从其中任意一个都可以推出其它两个。

本节提要

- 归纳：
  - 数学归纳法与强数学归纳法
  - 运用良序公理来证明
- 递归：
  - 递归定义
  - 结构归纳法与递归算法

递归



递归定义N上的函数

- 递归地定义自然数集合N上的函数
  - 基础步骤：指定这个函数在0处的值；
  - 递归步骤：给出从较小处的值来求出当前的值之规则。

- 举例，阶乘函数F(n)=n!的递归定义
  - F(0)=1
  - F(n)=n·F(n-1) for n>0

Fibonacci 序列

- Fibonacci 序列 $\{f_n\}$ 定义如下
  - $f_0 = 0,$
  - $f_1 = 1,$
  - $f_n = f_{n-1} + f_{n-2}$, 对任意$n \geq 2.$
- 其前几个数
  - $0, 1, 1, 2, 3, 5, 8, \cdots$
- 证明：对任意$n \geq 0$，$f_n = \dfrac{\alpha^n - \beta^n}{\alpha - \beta}$

其中，$\alpha = \dfrac{1+\sqrt{5}}{2}, \beta = \dfrac{1-\sqrt{5}}{2}.$

归纳证明: Fibonacci 序列

- 证明：当n=0,1时，陈述正确。

  对于k+1，

$$
\begin{aligned}
f_{k+1} &= f_k + f_{k-1} \\
&= \frac{\alpha^k - \beta^k}{\alpha - \beta} + \frac{\alpha^{k-1} - \beta^{k-1}}{\alpha - \beta} \\
&= \frac{\left(\alpha^k + \alpha^{k-1}\right) - \left(\beta^k + \beta^{k-1}\right)}{\alpha - \beta} \\
&= \frac{\alpha^{k+1} - \beta^{k+1}}{\alpha - \beta}.
\end{aligned}
$$

注意：$\alpha^2 = \alpha + 1$，且$\alpha^{n+1} = \alpha^n + \alpha^{n-1}$ 对任意$n \geq 1$.

类似可证：$n \geq 3$时，$f_n > \alpha^{n-2}$

递归定义集合

- 递归地定义集合
  - 基础步骤：指定一些初始元素；
  - 递归步骤：给出从集合中的元素来构造新元素之规则；
  - 排斥规则（只包含上述步骤生成的那些元素）默认成立

- 举例，正整数集合的子集S
  - $x \in S$
  - 若$x \in S$且$y \in S$，则 $x+y \in S$。

递归定义集合

- 字母表Σ上的字符串集合Σ^*
  - 基础步骤：$\lambda \in \Sigma^*$（λ表示空串）；
  - 递归步骤：若$\omega \in \Sigma^*$且$x \in \Sigma$，则$\omega x \in \Sigma^*$。

- 字符串的长度（Σ^*上的函数l）
  - 基础步骤：$l(\lambda)=0$;
  - 递归步骤：$l(\omega x) = l(\omega) + 1$, 若$\omega \in \Sigma^*$且$x \in \Sigma$

结构归纳法

- 关于递归定义的集合的命题，进行结构归纳证明
  - 基础步骤：证明对于初始元素来说，命题成立；
  - 递归步骤：针对生产新元素的规则，若相关元素满足命题，则新元素也满足命题
- 结构归纳法的有效性源于自然数上的数学归纳法
  - 对运用递归步骤的次数/步数进行归纳
  - 第0步（基础步骤）
  - 第k步推出k+1步（归纳步骤）

例

字符串的长度（Σ^*上的函数l）。

- 基础步骤：$l(\lambda)=0$;
- 递归步骤：$l(\omega x) = l(\omega) +1$, 若$\omega \in \Sigma^*$且$x \in \Sigma$

- 求证：$l(xy) = l(x) + l(y)$, x和y属于Σ^*。
- 证明：
  - 设$P(y)$表示：每当x属于Σ^*，就有$l(xy) = l(x) + l(y)$。
  - 基础步骤：每当x属于Σ^*，就有$l(x\lambda) = l(x) + l(\lambda)$。
  - 递归步骤：假设$P(y)$为真，a属于Σ, 要证$P(ya)$为真。
    - 即证：每当x属于Σ^*，就有$l(xya) = l(x) + l(ya)$
    - $P(y)$为真，$l(xy) = l(x) + l(y)$
    - $l(xya)= l(xy)+1= l(x) + l(y)+1=l(x) + l(ya)$

例（广义归纳法）

- 递归定义$a_{m,n}$
  - $a_{0,0} = 0$
  - $a_{m,n} = a_{m-1,n} + 1 \quad (n=0, m>0)$
  - $a_{m,n} = a_{m,n-1} + n \quad (n>0)$
- 归纳证明 $a_{m,n} = m+n(n+1)/2$

0	**1**	**3**
1	**2**	**4**
2	**3**	**5**

递归算法

An algorithm is called *recursive* if it solves a problem by reducing it to an instance of the same problem with smaller input.

- 举例：欧几里德算法

```
function gcd(a, b)  // a≥b≥0, a>0
 if b=0
      return a
 else
      return gcd(b, a mod b)
```

  - 递归算法的正确性 （数学归纳法）
  - 递归算法的复杂性 （时间、空间）

欧几里德算法的复杂性

- 拉梅定理: 设a和b是满足$a \geq b$的正整数。则欧几里德算法为求出$\gcd(a, b)$而使用除法的次数小于或等于b的十进制位数的5倍。$5(\lfloor \log_{10} b \rfloor + 1)$
- 令$r_0 = a, r_1 = b$.
- $r_0 = r_1 q_1 + r_2 \quad 0 \leq r_2 < r_1$
- $r_1 = r_2 q_2 + r_3 \quad 0 \leq r_3 < r_2$
- …
- $r_{n-1} = r_n q_n + r_{n+1} \quad 0 = r_{n+1} < r_n$
- $\gcd(a, b) = r_n$使用了n次除法

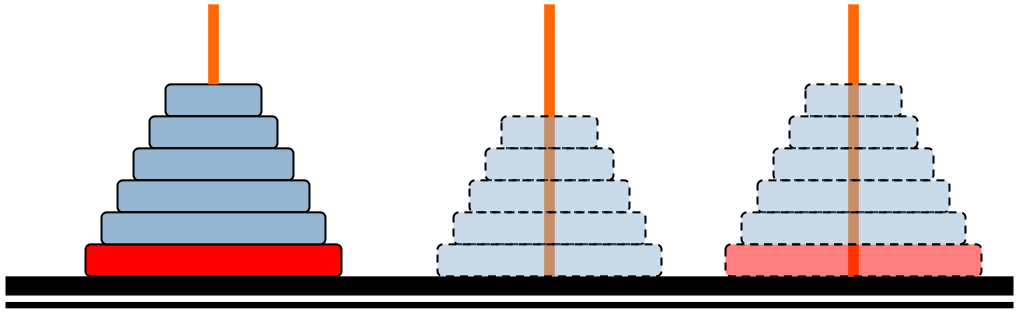
- $q_i \geq 1$ for $1 \leq i < n$
- $q_n \geq 2$ because $q_n = r_{n-1}/r_n > 1$
- $r_n \geq 1 = f_2, r_{n-1} \geq 2r_n \geq 2 = f_3$
- $b = r_1 \geq r_2 + r_3 \geq f_n + f_{n-1} = f_{n+1} > \alpha^{n-1}$
- $\log_{10} b > (n-1)\log_{10} \alpha$ for $n \geq 2$
- $\log_{10} \alpha > 1/5$

本节小结

- 归纳：
  - 数学归纳法与强数学归纳法
  - 运用良序公理来证明
- 递归：
  - 递归定义
  - 结构归纳法与递归算法

Thinking Recursively: Problem 1

- Towers of Hanoi（汉诺塔）
  - How many moves are needed to move all the disks from the first peg to the third peg, under the constraints that we can move only one disk at a time and never place a larger disk on top of a smaller one.



$$T(1) = 1$$
$$T(n) = 2T(n-1) + 1$$

Thinking Recursively: Problem 1

- Towers of Hanoi（汉诺塔）

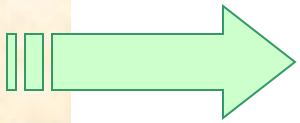
$T(n) = 2T(n-1) + 1$

$2T(n-1) = 4T(n-2) + 2$

$4T(n-2) = 8T(n-3) + 4$

……

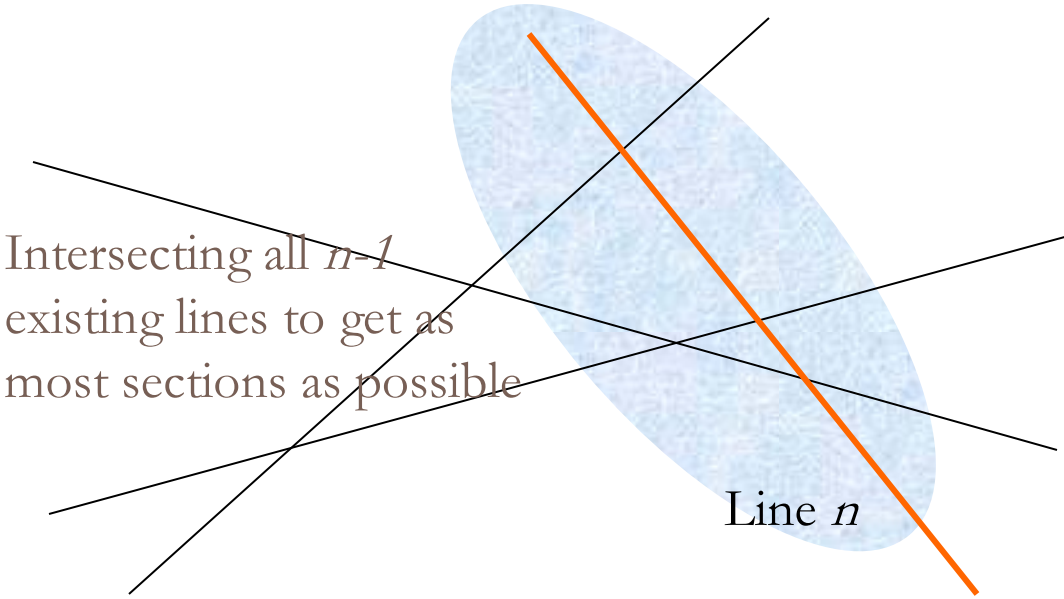
$2^{n-2}T(2) = 2^{n-1}T(1) + 2^{n-2}$



$$T(n) = 2^n - 1$$

Thinking Recursively: Problem 2

- Cutting the plane（平面切割）
  - How many sections can be generated at most by *n* straight lines with infinite length.



$$L(0) = 1$$
$$L(n) = L(n-1) + n$$

Thinking Recursively: Problem 2

- Cutting the plane（平面切割）

$$
\begin{aligned}
L(n) &= L(n-1)+n \\
&= L(n-2)+(n-1)+n \\
&= L(n-3)+(n-2)+(n-1)+n \\
&= \cdots\cdots \\
&= L(0)+1+2+\cdots+(n-2)+(n-1)+n
\end{aligned}
$$

$$L(n) = n(n+1)/2 + 1$$

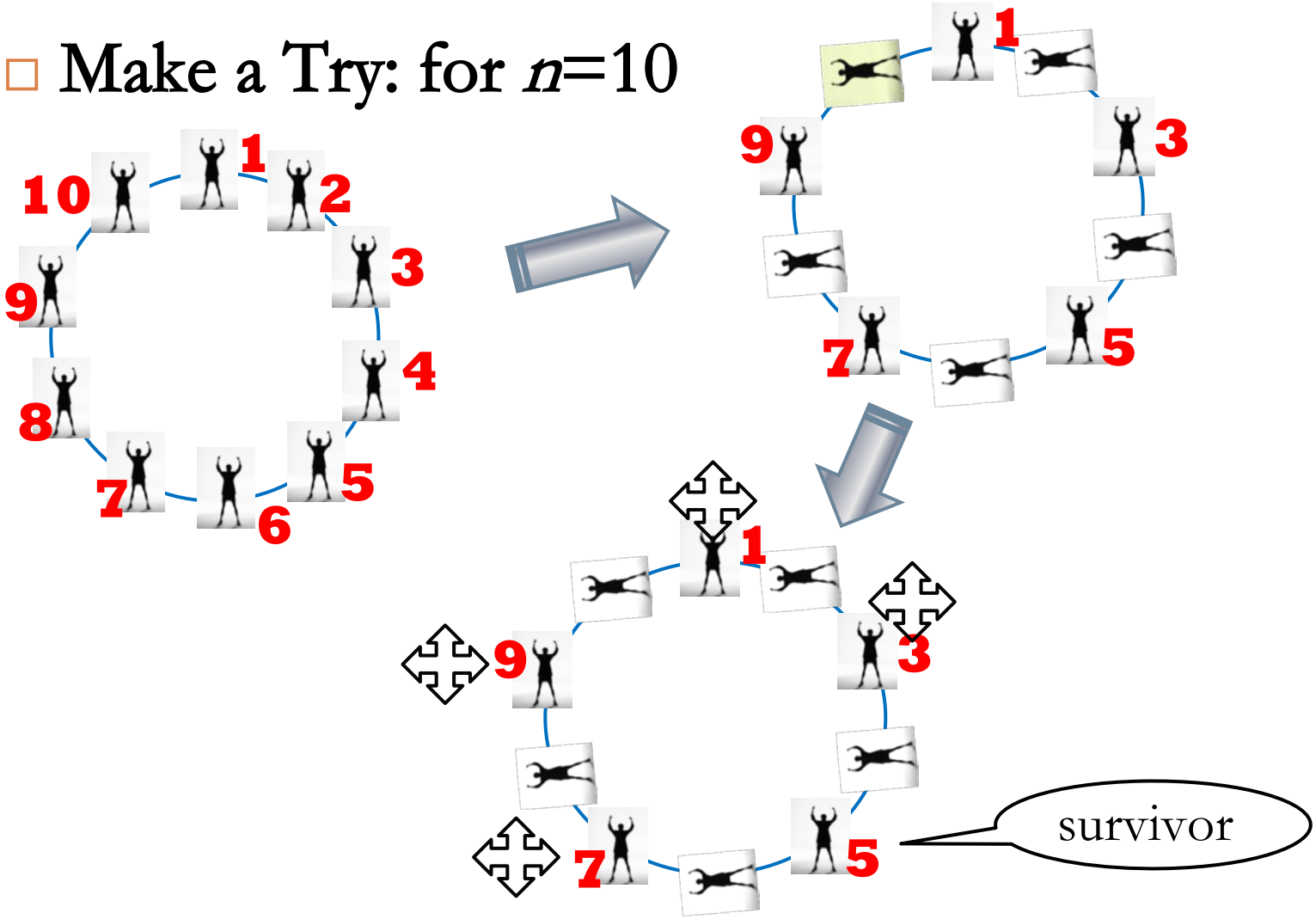
Thinking Recursively: Problem 3

- Josephus Problem: Live or die, it's a problem!
  - Legend has it that Josephus wouldn't have lived to become famous without his mathematical talents. During the Jewish Roman war, he was among a band of 41 Jewish rebels trapped in a cave by the Romans. Preferring suicide to capture, the rebels decided to form a circle and, proceeding around it, to kill every third remaining person until no one was left. But Josephus, along with an unindicted co-conspirator, wanted none of this suicide nonsense; so he quickly calculated where he and his friend should stand in the vicious circle.

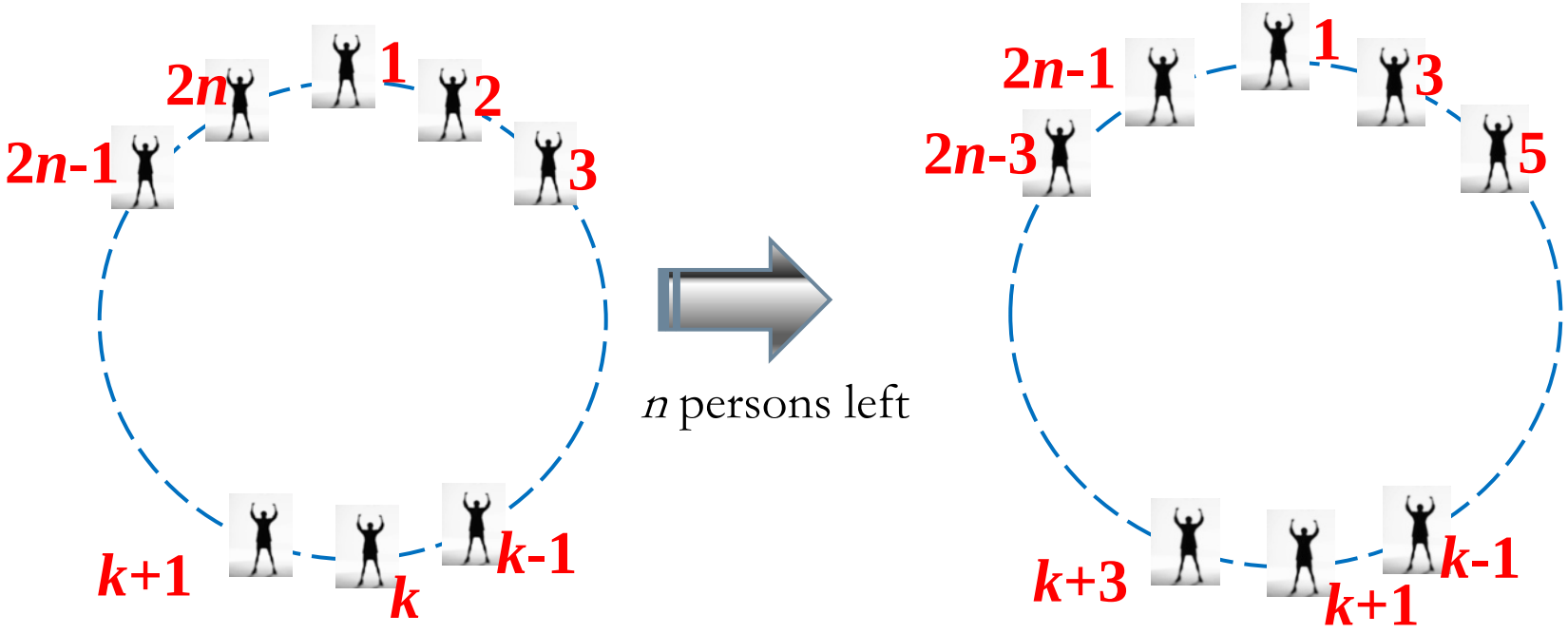
We use a simpler version: "every second..."

Thinking Recursively: Problem 3

- Make a Try: for n=10



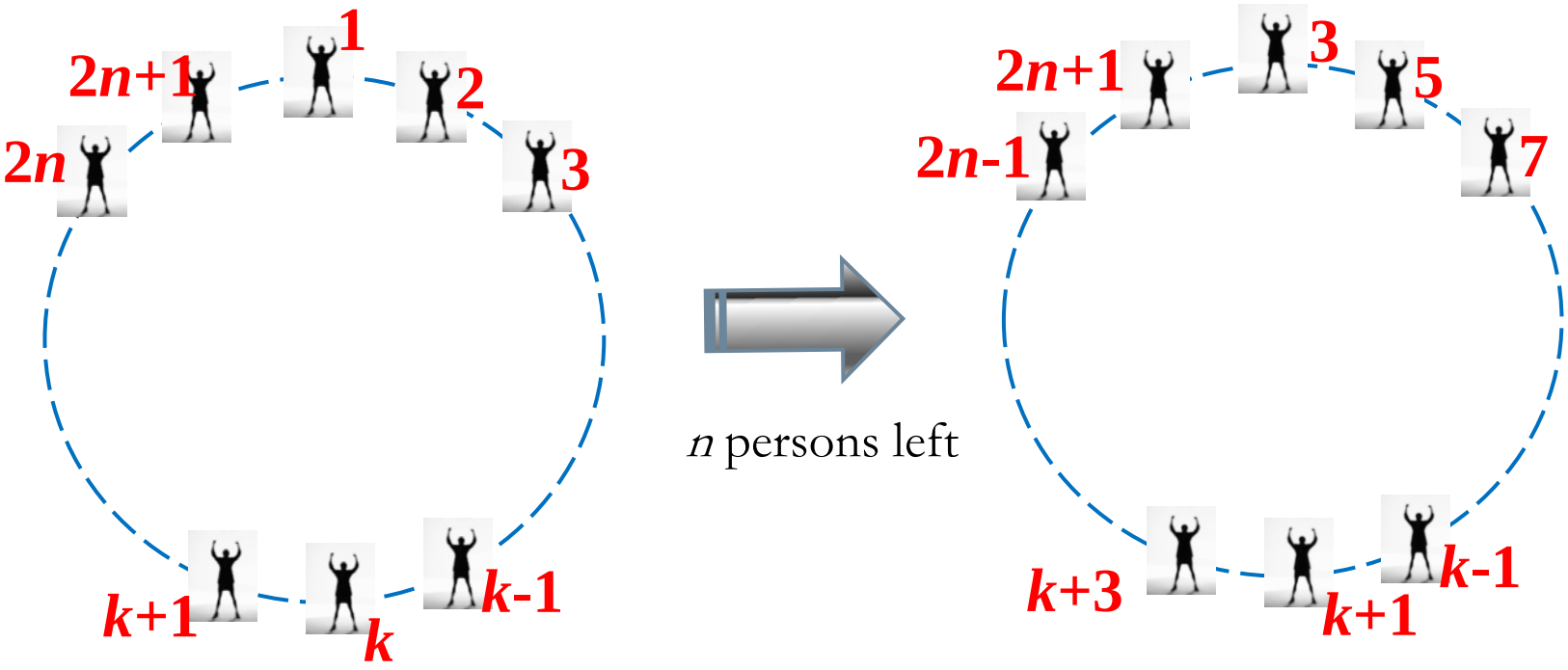
Thinking Recursively: Problem 3



一轮游戏后幸存者的编号为k，游戏开始前编号为 newnumber (k)

And the newnumber(*k*) is **2*k*-1**

Thinking Recursively: Problem 3



And for this time, the newnumber(k) is $2k+1$

Thinking Recursively: Problem 3

- Solution in Recursive Equations

$$J(1) = 1;$$
$$J(2n) = 2J(n) - 1, \quad \text{for } n \geq 1;$$
$$J(2n+1) = 2J(n) + 1, \quad \text{for } n \geq 1.$$

- Some explicit solutions

Can you find the pattern?

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
J(n)	1	1	3	1	3	5	7	1	3	5	7	9	11	13	15	1

Thinking Recursively: Problem 3

If we write *n* in the form $n = 2^m + l$,
(where 2^m is the largest power of 2 not exceeding *n* and *l* is what's left),
the solution to our recurrence seems to be:

$$J(2^m + l) = 2l + 1, \quad \text{for } m \geq 0 \text{ and } 0 \leqslant l < 2^m.$$

As an example: J(100) = J(64+36) = 36*2+1 = 73

Linear Homogeneous Recurrence Relation

$$a_n = r_1 a_{n-1} + r_2 a_{n-2} + \cdots + r_k a_{n-k}$$

is called linear homogeneous relation of degree *k*.

$c_n = (-2)c_{n-1}$

$f_n = f_{n-1} + f_{n-2}$

Yes

$a_n = a_{n-1} + 3$

$g_n = g_{n-1}^2 + g_{n-2}$

No

Characteristic Equation

- For a linear homogeneous recurrence relation of degree k

$$a_n = r_1 a_{n-1} + r_2 a_{n-2} + \cdots + r_k a_{n-k}$$

the polynomial of degree k

$$x^k = r_1 x^{k-1} + r_2 x^{k-2} + \cdots + r_k$$

is called its characteristic equation.

- The characteristic equation of linear homogeneous recurrence relation of degree 2 is:

$$x^2 - r_1 x - r_2 = 0$$

Solution of Recurrence Relation

- If the characteristic equation $x^2 - r_1 x - r_2 = 0$ of the recurrence relation $a_n = r_1 a_{n-1} + r_2 a_{n-2}$ has two distinct roots s_1 and s_2, then we have the explicit formula for the sequence

$$a_n = u s_1^n + v s_2^n$$

where u and v depend on the initial conditions.

Proof of the Solution

Remember the equation : $x^2 - r_1 x - r_2 = 0$

We need prove that : $us_1^n + vs_2^n = r_1 a_{n-1} + r_2 a_{n-2}$

$$
\begin{aligned}
& us_1^n + vs_2^n = us_1^{n-2} s_1^2 + vs_2^{n-2} s_2^2 \\
& = us_1^{n-2}(r_1 s_1 + r_2) + vs_2^{n-2}(r_1 s_2 + r_2) \\
& = r_1 us_1^{n-1} + r_2 us_1^{n-2} + r_1 vs_2^{n-1} + r_2 vs_2^{n-2} \\
& = r_1(us_1^{n-1} + vs_2^{n-1}) + r_2(us_1^{n-2} + vs_2^{n-2}) \\
& = r_1 a_{n-1} + r_2 a_{n-2}
\end{aligned}
$$