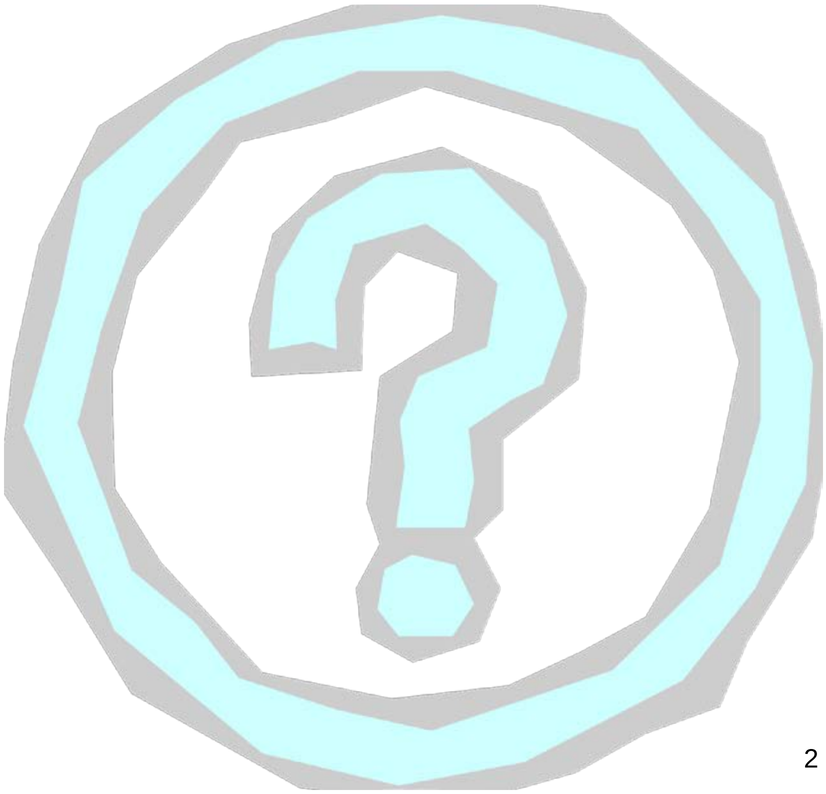
归纳与递归

离散数学—归纳与递归

南京大学计算机科学与技术系

内容提要

- 递归定义
- 结构归纳法
- 递归算法



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
  - $F(0)=1$
  - $F(n)=n \cdot F(n-1)$ 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, …**
- **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

- **验证：当n=0,1时，陈述正确。**
- **对于k+1,** $f_{k+1} = f_k + f_{k-1}$

$$= \frac{\alpha^k - \beta^k}{\alpha - \beta} + \frac{\alpha^{k-1} - \beta^{k-1}}{\alpha - \beta}$$

$$= \frac{\left(\alpha^k + \alpha^{k-1}\right) - \left(\beta^k + \beta^{k-1}\right)}{\alpha - \beta}$$

$$= \frac{\alpha^{k+1} - \beta^{k+1}}{\alpha - \beta}.$$

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

归纳证明: Fibonacci 序列

- 证明： 当$n \geq 3$时， $f_n > \alpha^{n-2}$
- 基础步骤
  - 当n=3时， $f_3 = 2 > \alpha = (1+\sqrt{5})/2$
  - 当n=4时， $f_4 = 3 > \alpha^2 = (3+\sqrt{5})/2$
- 归纳步骤
  - $n \geq 4$时, $f_{n+1} = f_n + f_{n-1} > \alpha^{n-2} + \alpha^{n-3} = \alpha^{n-1}$.
- 所以， 当$n \geq 3$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
  - $2 \in S$
  - 若$x \in S$且$y \in S$，则 $x+y \in S$。

递归定义（举例）

- 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$x \in \Sigma$

递归定义（举例）

- Σ^*上的字符串连接运算。
  - 基础步骤：若$\omega \in \Sigma^*$，则 $\omega \cdot \lambda = \omega$;
  - 递归步骤：若$\omega_1 \in \Sigma^*$ 且$\omega_2 \in \Sigma^*$ 以及$x \in \Sigma$，则 $\omega_1 \cdot (\omega_2\, x) = (\omega_1 \cdot \omega_2)\, x$ 。
  - // $\omega_1 \cdot \omega_2$通常也写成$\omega_1\, \omega_2$

递归定义（举例）

- **复合命题的合式公式。**
  - **基础步骤：T, F, *p*都是合式公式，其中*p*是命题变元;**
  - **递归步骤：若E和F是合式公式，则 ($\neg$E)、(E$\wedge$F)、(E$\vee$F)和(E$\rightarrow$F) 都是合式公式。**

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

结构归纳法（举例）

- $l(xy) = l(x) + l(y)$, 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：每当x属于 Σ^*，就有$l(xya) = l(x) + l(ya)$
    - $P(y)$为真，$l(xy) = l(x) + l(y)$
    - $l(xya)= l(xy)+1= l(x) + l(y)+1=l(x) + l(ya)$

广义结构归纳法（举例）

- N×N是良序的（字典序）
- 递归定义$a_{m,n}$
  - $a_{0,0} = 0$
  - $a_{m,n} = a_{m-1,n} + 1 \ \ (n=0, m>0)$
  - $a_{m,n} = a_{m,n-1} + n \ \ (n>0)$
- 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（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
- 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$r_0=a$, $r_1=b$.
- $r_0 = r_1 q_1 + r_2 \qquad 0 \leq r_2 < r_1$
- $r_1 = r_2 q_2 + r_3 \qquad 0 \leq r_3 < r_2$
- …
- $r_{n-1} = r_n q_n + r_{n+1} \quad 0 = r_{n+1} < r_n$
- gcd(a, b) = r_n使用了n次除法

- Let $r_0=a$, $r_1=b$.
- $q_i \geq 1$ 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 $n \geq 2$
- $\log_{10} \alpha > 1/5$

递归算法的设计

- $\boldsymbol{a^n}$
- $\boldsymbol{b^n \bmod m}$

作业

- 教材[4.3, 4.4]
  - P232-236：24, 32, 34, 47, 52
  - P244-245：36, 37