函数

离散教学小组

School of Computer
Wuhan University

Circle Limit IV — Escher

函数的定义

Definition (函数，function(map, mapping))

设f是集合X到Y上的关系($f \subseteq X \times Y$)，f是函数，iff, f满足下述两条件：

1. 完全性：$\forall x \in X\ \exists y \in Y$, such that, $\langle x, y\rangle \in f$;
2. 多对一：if $\langle x, y\rangle \in f \wedge \langle x, z\rangle \in f$, then $y = z$;

集合X和Y分别称为函数f的定义域(domain)和陪域(codomain)，与$x \in X$有关系f的$y \in Y$记为：$f(x)$.

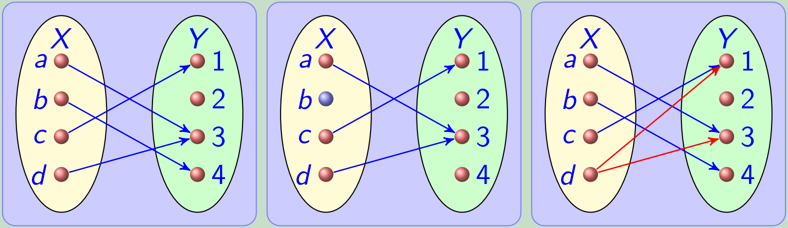
Remark

对上述两条件的任意一个的破坏，将不能构成函数：

1. *非完全的：* $\exists x \in X \wedge \forall y \in Y, \langle x, y\rangle \notin f$:
   表示函数在有些点没有定义，将满足条件② 不满足条件①的关系称为部分函数(partial);
2. *一对多：if* $\exists x \in X \wedge y, z \in Y \wedge y \neq z \wedge \langle x, y\rangle \in f \wedge \langle x, z\rangle \in f$
   表示函数在有些点可能对应多值，将满足条件①不满足条件② 的关系称为多值函数(multivalued)。

图例

Example



(a) 函数 (b) 部分函数 (c) 多值函数

相关定义

Definition (函数集合)

- 记$Y^X \triangleq \{ f : X \longrightarrow Y \}$为所有的从$X$到$Y$的函数集合;
- if X和Y为有限集合，则$|Y^X| = |Y|^{|X|}$;
- $|Y|^{|X|} < 2^{|x| \times |y|}$, $\because Y^X \subsetneq \mathscr{P}(X \times Y)$.

Definition (相等)

称两函数$f, g : X \longrightarrow Y$相等(记为$f = g$), iff，对应的函数关系在集合意义下相等，即$\forall x\ f(x) = g(x)$.

Example

- $\left|\{0, 1\}^{\{0,1\}^n}\right| = 2^{2^n}$;
- $F \Leftrightarrow G$ iff $I(F) = I(G)$;

函数的象和逆象

Definition (象，image; 逆象，inverse image)

设$f : X \longrightarrow Y$是X到Y上的函数，$A \subseteq X$, $B \subseteq Y$:

- A在f下的象:
  $f(A) \triangleq \{ y \mid \exists x \in A \wedge y = f(x) \} = \{ f(x) \mid x \in A \}$;
- B在f下的逆象:
  $f^{-1}(B) \triangleq \{ x \mid \exists y \in B \wedge y = f(x) \} = \{ x \mid f(x) \in B \}$;
- $f(X)$称为函数f的值域(range).

Definition

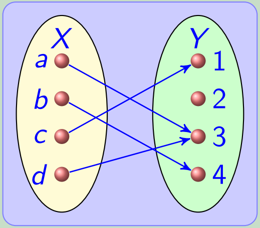
- $f : \mathbb{R} \longrightarrow \mathbb{R}$, if f是连续函数，则对任意的开区间$]a, b[$有$f(]a, b[)$也是开区间;
- 重言式集合$\mathcal{T} = \mathscr{I}^{-1}(\{ \mathbb{T} \})$.

Example

Remark

f^{-1}有两重含义，求逆象和反函数，求逆象的作用对象只能是集合: $\cancel{f^{-1}(b)}$; 只有当f有反函数存在时，f^{-1}才能作用元素。

Example



- $f(\{a, d\}) = \{3\}$;
- $f(\{d\}) = \{3\}$;
- $f^{-1}(\{3\}) = \{a, d\}$;
- $f^{-1}(\{2\}) = \varnothing$;

相关性质

Propostion

设$f: X \longrightarrow Y$是X到Y上的函数，$A \subseteq X$, $B \subseteq Y$，则：

- $A \subseteq f^{-1}(f(A))$;
- $f(f^{-1}(B)) \subseteq B$.

Proof.

1. $\forall x \in A$, 则$f(x) \in f(A)$,
   $\therefore x \in f^{-1}(f(A)) = \{x' \mid f(x') \in f(A)\}$;
2. $\forall y \in f(f^{-1}(B))$, 则$\exists x \in f^{-1}(B) \wedge y = f(x)$,
   而$\because x \in f^{-1}(B)$, $\therefore f(x) \in B$, 即$y \in B$.

$\square$

Remark

直观上，求象：对集合缩小；求逆象: 对集合放大。

常用的函数

Description

1. $\mathbb{1}_X : X \longrightarrow X,\ x \longmapsto x$ 恒等函数;
2. $b : X \longrightarrow Y,\ x \longmapsto b$ 常数函数;
3. $s : \mathbb{N} \longrightarrow \mathbb{N},\ n \longmapsto n+1$后继函数;
4. $f : X_1 \times X_2 \times \cdots \times X_n \longrightarrow Y$ n元函数;
5. $p_i : X_1 \times X_2 \times \cdots \times X_n \longrightarrow X_i,\ \langle x_1, x_2, \ldots, x_n \rangle \longmapsto x_i$ 投影函数;
6. $X \longrightarrow \mathscr{P}(X \times Y),\ x \longmapsto \{x\} \times Y$ 截痕函数。

函数的合成

Definition (合成函数(复合), Composite function)

设$f: X \longrightarrow Y$是X到Y上的函数，$g: Y \longrightarrow Z$是Y到Z上的函数，f和g的合成$g \circ f$：$X \longrightarrow Z$，$x \longmapsto g(f(x))$也是函数，称为合成函数 (注意：在写法上与关系的合成相反)。

Definition (交换图，Commutative Diagram)

$$
\begin{tikzcd}
X \arrow[r, "f"] \arrow[dr, "g \circ f"'] & Y \arrow[d, "g"] \\
 & Z
\end{tikzcd}
$$

相关性质

Propostion

设$f : X \longrightarrow Y$, $g : Y \longrightarrow Z$, $h : Z \longrightarrow W$:

- $\mathbb{1}_Y \circ f = f \circ \mathbb{1}_X = f$;
- $h \circ (g \circ f) = (h \circ g) \circ f$ (结合律).

Notation

设$f : X \longrightarrow X$:

- $f^0 = \mathbb{1}_A$;
- $f^{n+1} = f \circ f^n$;
- 和关系一样：$\forall m, n \in \mathbb{N}\ f^m \circ f^n = f^{m+n}$; $(f^m)^n = f^{mn}$.

单射，满射和双射

Definition

设$f: X \longrightarrow Y$:

- if $f(X) = Y$, 称f为满射(onto)
- if $\forall x, x' \in X\ f(x) = f(x') \to x = x'$ (或者$\forall xx' \in X\ x \neq x' \to f(x) \neq f(x')$), 称$f$为单射(one to one);
- if f既是单射也是满射，则称f为双射(bijection).

Example

设$|X| = m, |Y| = n$:

- $\mathbb{1}_X$是双射;
- f是单射，iff , $|f(X)| = m$, 因此$|X| \leqslant |Y|$;
- f是满射，则, $|Y| = |f(X)| \leqslant |X|$, 因此$|Y| \leqslant |X|$;
- f是双射，则$|X| = |Y|$.

Examples

Example

设$|X|=m, |Y|=n$:

- $\{1,2,\ldots,n\}$上的双射称为置换;
  记$P_n=\{\{1,2,\ldots,n\}$所有置换的集合$\}$, 则$|P_n|=n!$;
- 设$n\geqslant m$ 则X到Y上的单射共有$C_n^m m!$个;
- 设$n\leqslant m$ 则X到Y上的满射的个数等于m个元素的集合共有多少个n分区的个数.

相关性质

Propostion

设$f : X \longrightarrow Y, g : Y \longrightarrow Z$:

1. 设f, g是单射, 则$g \circ f$也是单射;
2. 设f, g是满射, 则$g \circ f$也是满射;
3. 设f, g是双射, 则$g \circ f$也是双射。

①的证明.

1. $x, x' \in X$, if $g(f(x)) = g(f(x'))$;
2. $\because g$是单射, $\therefore f(x) = f(x')$;
3. 又f是单射, $\therefore x = x'$.

$\square$

单射满射充要条件

Propostion

设$f : X \longrightarrow Y$:

1. f是单射，iff，$\exists g : Y \longrightarrow X \wedge g \circ f = \mathbb{1}_X$，称$g$为$f$的*左逆元*；
2. f是满射，iff，$\exists g : Y \longrightarrow X \wedge f \circ g = \mathbb{1}_Y$，称$g$为$f$的*右逆元*。

①的证明.

$\Longleftarrow$ 设$f(x) = f(x')$，则$g(f(x)) = g(f(x'))$，即$x = x'$；所以f是单射；

$\Longrightarrow$ 设$x_0 \in X$，构造$g : Y \longrightarrow X$:

$$g(y) = \begin{cases} x & if\ y \in f(X) \wedge f(x) = y; \\ x_0 & if\ y \notin f(X); \end{cases}$$

$\because g$是单射，$\exists! x \in X \wedge f(x) = y$, if $y \in f(X)$;

$\therefore g$ is well-defined, And $g \circ f(x) = g(f(x)) = x$;

从构造上左右逆元不唯一！

□

双射充要条件

Propostion

设$f: X \longrightarrow Y$, f是双射，iff，$\exists! g: Y \longrightarrow X \wedge g \circ f = \mathbb{1}_X \wedge f \circ g = \mathbb{1}_Y$，称$g$为$f$的逆元，并记该逆元为$f^{-1}$.

Proof.

$\Longrightarrow$ 设f是双射，由上命题$\exists g, g': Y \longrightarrow X$ $g \circ f = \mathbb{1}_X \wedge f \circ g' = \mathbb{1}_Y$;

$$\begin{aligned}\therefore\quad & g \\ &= g \circ \mathbb{1}_Y \\ &= g \circ (f \circ g') \\ &= (g \circ f) \circ g' \\ &= \mathbb{1}_X \circ g' \\ &= g';\end{aligned}$$

$\therefore g = g'$，即g存在并且唯一确定的；

$\Longleftarrow$ 由上命题得知：f既是单射也是满射，即f是双射。

$\square$

Examples

Example

- 设$f(x) = \lg(x)$, 则$f^{-1}(x) = \mathrm{e}^x \therefore \mathrm{e}^{\lg x} = x \wedge \lg(\mathrm{e}^x) = x$:

$$
\begin{aligned}
& (\mathrm{e}^{\lg x})' = x' \\
\Longrightarrow \quad & \mathrm{e}^{\lg x} \cdot (\lg x)' = 1 \\
\Longrightarrow \quad & x \cdot (\lg x)' = 1 \\
\Longrightarrow \quad & (\lg x)' = 1/x
\end{aligned}
$$

- 设f, g都是双射，则$(g \circ f)^{-1} = f^{-1} \circ g^{-1}$:

$$
\begin{aligned}
\because \quad & (f^{-1} \circ g^{-1}) \circ (g \circ f) = \mathbb{1}_X \\
\wedge \quad & (g \circ f) \circ (f^{-1} \circ g^{-1}) = \mathbb{1}_Z
\end{aligned}
$$

Examples

Example

$f: X \longrightarrow Y$，f是单射，iff，$\forall A \subseteq X\ f^{-1}(f(A)) = A$;

$\Longrightarrow$

1. $A \subseteq f^{-1}(f(A))$ ✓
2. $f^{-1}(f(A)) \subseteq A$?
3. $\forall x \in f^{-1}(f(A))$，则$f(x) \in f(A)$;
4. $\therefore \exists x' \in A \wedge f(x) = f(x')$;
5. 而f是单射，$\therefore x = x' \in A$.

$\Longleftarrow$

1. 设$f(x) = f(x')$，证明$x = x'$;
2. 令$A = \{x\}$，则$f^{-1}(f(\{x\})) = f^{-1}(\{f(x)\}) = \{x\}$;
3. 而$f(x') \in \{f(x)\}$;
4. $\therefore x' \in f^{-1}(\{f(x)\}) = f^{-1}(f(\{x\})) = \{x\}$;
5. $\therefore x = x'$.

Examples

Example

$f: X \longrightarrow Y$，f是满射，iff，$\forall B \subseteq Y\ f(f^{-1}(B)) = B$.

$\Longrightarrow$

1. $f(f^{-1}(B)) \subseteq B$ ✓
2. $B \subseteq f(f^{-1}(B))$?
3. $\forall y \in B$;
4. $\because f$是满射，$\exists x \in X\ f(x) = y$;
5. $\therefore x \in f^{-1}(B)$;
6. $\therefore f(x) \in f(f^{-1}(B))$;
7. 即$y \in f(f^{-1}(B))$.

$\Longleftarrow\ f(X) = Y$?

1. 令$B = Y$，则$Y = f(f^{-1}(Y))$;
2. 又$f^{-1}(Y) \subseteq X$;
3. $\therefore f(f^{-1}(Y)) \subseteq f(X)$;
4. $\therefore Y = f(f^{-1}(Y)) \subseteq f(X) \subseteq Y$, $\therefore Y = f(X)$.

Example

Example

$f:\mathbb{N}\longrightarrow\mathbb{N}$递归定义如下，求$f$的解析式。

1. $f(0)=3$;
2. $f(n+1)=2f(n)+3$.

Solution.

$$\begin{aligned}
f(n) &= 2*f(n-1)+3\\
&= 2*(2f(n-2)+3)+3\\
&= 2^2*f(n-2)+2*3+3\\
&= 2^2*(2f(n-3)+3)+2*3+3\\
&= 2^3*f(n-3)+2^2*3+2*3+3\\
&\quad \cdots\cdots\\
&= 2^n*f(0)+2^{n-1}*3+\cdots+2^2*3+2*3+3\\
&= 3*(2^n+2^{n-1}+\cdots+2^2+2+1)\\
&= 3*(2^{n+1}-1)
\end{aligned}$$

$\square$

对应的程序

Recursion

```
int f(int n)
{
  if (n < 0) error();
  if (n == 0) return 3;
  return 2 * f(n-1) + 3;
}
```

For-loops

```
int f(int n)
{
  int result = 3, i;
  if (n < 0) error();
  for (i = 1; i <= n; i++)
    result = 2 * result + 3;
  return result;
}
```

Example

Example

Fibonacci序列递归定义如下，求其解析式。

1. $f(0) = 0,\ f_1 = 1;$
2. $f_n = f_{n-1} + f_{n-2}\ \ (n \geqslant 2).$

Solution.

1. 求方程$r^2 = r + 1$的两个根r_1和r_2;
2. 设$f_n = \alpha_1 r_1^n + \alpha_2 r_2^n$;
3. 则有：$f_n = f_{n-1} + f_{n-2}$;
4. 利用条件①可求出系数α_1和α_2;
5. $f_n = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n.$

□

对应的程序

Recursion

```
 int f(int n)
{
  if (n == 0) return 0;
  if (n == 1) return 1;
  return f(n-2) + f(n-1);
}
```

For-loops

```
int f(int n)
{
  int x = 0;  /* f(n-2) */
  int y = 1;  /* f(n-1) */
  if (n == 0) return 0;
  if (n < 0) error();
  for (i = 1; i <= n-1; i++) {
    int z = x + y;
    x = y;
    y = z;
    }
  return y;
}
```

Euclid 算法

Example (Greatest Common Divisor)

$\mathrm{gcd}: \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$可递归定义如下:

1. $\mathrm{gcd}(0, n) = n$;
2. $\mathrm{gcd}(m, n) = \mathrm{gcd}((n \bmod m), m)$.

Proof.

1. if $(k \mid m) \wedge (k \mid n)$,
2. then $m = kp \wedge n = kq$;
3. 设$n \bmod m = t$, 则$n = rm + t$;
4. $\therefore t = k(q - rp)$; 即$k \mid t$; $\therefore \mathrm{gcd}(m, n) \leqslant \mathrm{gcd}(n \bmod n, m)$;
5. 设$(k \mid t) \wedge (k \mid m)$; then $\exists p, q\ t = kp \wedge m = kp$;
6. $\therefore n = k(rq + p)$, 即$t \mid n$; $\therefore \mathrm{gcd}(m, n) \geqslant \mathrm{gcd}(n \bmod n, m)$.

□

对应的程序

Tail Recursion

```
 int gcd(int m, int n)
{
  if (m == 0) return n;
  return gcd( n % m, m);
}

Ex:
  gcd(18, 12)
    ↳gcd(12, 18)
       ↳gcd(6, 12)
          ↳gcd(0, 6)
             ↳6
```

While-loops

```
int gcd(int m, int n)
{
  int tmp;
  while (m != 0) {
  tmp = m;
  m =  n % m;
  n = tmp;
  }
  return n;
}
```

List上的递归定义函数

Example (length)

$\mathrm{length} : \Sigma^* \longrightarrow \mathbb{N}$可递归定义如下:

1. $\mathrm{length}(\varepsilon) = 0$;
2. $\mathrm{length}(a \cdot s) = 1 + \mathrm{length}(s)$.

Example (length尾递归定义)

$f : \mathbb{N} \times \Sigma^* \longrightarrow \mathbb{N}$可递归定义如下:

1. $f(m, \varepsilon) = m$;
2. $f(m, a \cdot s) = f(m + 1, s)$.

So, $\mathrm{length}(s) = f(0, s)$

对应的程序

Tail Recursion

```
int f(int m, Σ List s)
{
  if ( s == ε) return m;
  return f(m + 1, tl(s));
    /* tl is destructor of Σ List */
       tl(a·s) = s */
}

int length(Σ List s)
{
  return f(0, s);
}
```

While-loops

```
int length(Σ List s)
{
  int m = 0;
  while (s != ε) {
  m = m + 1;
  s = tl(s);
  }
  return m;
}

int length(char * s)
{
  int tmp = 0;
  while(*s++) tmp++;
  return tmp;
}
```

List上的递归定义函数

Example (sum)

$\mathrm{sum} : \mathbb{N}^* \longrightarrow \mathbb{N}$可递归定义如下：

1. $\mathrm{sum}(\varepsilon) = 0$;
2. $\mathrm{sum}(a \cdot s) = a + \mathrm{sum}(s)$.

Example (sum尾递归定义)

$f : \mathbb{N} \times \mathbb{N}^* \longrightarrow \mathbb{N}$可递归定义如下：

1. $f(m, \varepsilon) = m$;
2. $f(m, a \cdot s) = f(m + a, s)$.

So, $\mathrm{sum}(s) = f(0, s)$.

对应的程序

Tail Recursion

```
 int f(int m, int List s)
{
  if ( s == ε) return m;
  return f(m + hd(s), tl(s));
    /* tl, hd are destructors
       of int List:
       tl(a·s) = s
       hd(a·s) = a  */
}

int sum (int List s)
{
  return f(0, s);
}
```

While-loops

```
 int sum(int List s)
{
  int m = 0;
  while (s != ε) {
    m = m + hd(s);
    s = tl(s);
    }
  return m;
}
```

Example

Example

Ackerman函数递归定义如下：

$$A(m,n) = \begin{cases} n+1 & \text{if } m=0 \\ A(m-1,1) & \text{if } m>0 \wedge n=0 \\ A(m-1,A(m,n-1)) & \text{if } m>0 \wedge n>0 \end{cases}$$

Description

1. *处处有定义;*
2. *m足够小时，增长缓慢，$m \geqslant 4$时呈指数增长($A(4,2) \doteq 2*10^{19728}$);*
3. *非原始递归函数(primitive recursive function);*
4. *不能够用while-loops表达;*
5. *由于其深度递归性(deep recursion)，该函数用于测试编译器对递归的优化性能。*

A(4,3)的计算

```
A(4, 3) = A(3, A(4, 2))
        = A(3, A(3, A(4, 1)))
        = A(3, A(3, A(3, A(4, 0))))
        = A(3, A(3, A(3, A(3, 1))))
        = A(3, A(3, A(3, A(2, A(3, 0)))))
        = A(3, A(3, A(3, A(2, A(2, 1)))))
        = A(3, A(3, A(3, A(2, A(1, A(2, 0))))))
        = A(3, A(3, A(3, A(2, A(1, A(1, 1))))))
        = A(3, A(3, A(3, A(2, A(1, A(0, A(1, 0)))))))
        = A(3, A(3, A(3, A(2, A(1, A(0, A(0, 1)))))))
        = A(3, A(3, A(3, A(2, A(1, A(0, 2))))))
        = A(3, A(3, A(3, A(2, A(1, 3)))))
        = A(3, A(3, A(3, A(2, A(0, A(1, 2))))))
        = A(3, A(3, A(3, A(2, A(0, A(0, A(1, 1)))))))
        = A(3, A(3, A(3, A(2, A(0, A(0, A(0, A(1, 0))))))))
        = A(3, A(3, A(3, A(2, A(0, A(0, A(0, A(0, 1))))))))
        = A(3, A(3, A(3, A(2, A(0, A(0, A(0, 2)))))))
        = A(3, A(3, A(3, A(2, A(0, A(0, 3))))))
        = A(3, A(3, A(3, A(2, A(0, 4)))))
        = A(3, A(3, A(3, A(2, 5))))
        = ...
        = A(3, A(3, A(3, 13)))
        = ...
        = A(3, A(3, 65533))    /* A(3, 65533) = 2^65533-3 */
        = ...
```

对应的程序

Recursion

```
int ack(int m, int n)
{
  if (m == 0) return n+1;
  if (n == 0) return ack(m-1, 1);
  return ack(m-1, ack(m, n-1))
}
  /* No linear recursion */
```

Partially while-loops

```
int ack(int m, int n)
{
  while (m != 0) {
    if (n == 0)
      n = 1;
    else
      n = ack(m, n-1);
    m = m - 1;
  }
  return n+1;
}
```

高阶函数

Example (高阶函数)

- 设$C(\mathbb{R})$是实数上连续函数的集合：$g : C(\mathbb{R}) \longrightarrow \mathbb{R}^{\mathbb{R}^2}$，
$f \longmapsto (\langle x, y\rangle \longmapsto \int_x^y f(t)dt)$，则$(g(\sin))(0,1)$是sin在$[0,1]$区间上的积分；
- 设$\mathrm{fold_left} : X^{X\times\Sigma} \longrightarrow X^{X\times\Sigma^*}$，
$f \longmapsto (\langle x, a_1a_2\ldots a_n\rangle \longmapsto f(\cdots(f(x,a_1),a_2)\ldots))$;
设$\Sigma = \mathbb{Z}; X = \mathbb{Z}, f(x,y) = x + y$，则：
$$\mathrm{sum}(a_1a_2\cdots a_n) = (\mathrm{fold\_left}(f))(0, a_1a_2\cdots a_n)$$
设$f(x,y) = 1 + x$，则：
$$\mathrm{length}(a_1a_2\cdots a_n) = (\mathrm{fold\_left}(f))(0, a_1a_2\cdots a_n)$$

小结

Remark

- *函数能够递归定义的首要条是定义域必须有一个递归结构;*
- *一般是按照定义域集合的归纳定义，对该集合中的每个元素进行析构;*
- *recursions $\neq$ loops;*
- *tail recursion can be transform to while-loop;*
- *recursion: simple & clear, but expensive in runtime;*
- *loop: complex & unclear, but more efficient.*

本章小节

1. 函数的基本概念
   - 函数的定义
   - 函数的象和逆象
   - 函数的合成
   - 单射，满射和双射
2. 函数的分解
   - Examples
   - 函数的标准分解
3. 函数的递归定义
   - 自然数集合上的递归函数
   - Euclid算法与尾递归
   - List集合上的递归定义函数
   - Ackerman函数
   - 高阶函数