

Discrete

Mathematics - P112

克努特箭头记法 (Knuth's up-arrow notation), 是一种用于表示大整数的方法 (method of notation for very large integer)

本质上可以认为克努特箭头是一种运算符, 且可以递归地定义。

首先根据超运算的阶考虑克努特箭头的运用

超运算

克努特箭头

自继运算

$$a^1 = a + 1$$

加法

$$a+n = \underbrace{(-ca')^1 \cdots 1}_{n \text{ 次}}^1$$

乘法

$$a * n = \underbrace{0 + a + a + \cdots + a}_{n \text{ 次}}$$

幂

$$a^n = \underbrace{1 \times a \times a \times \cdots \times a}_{n \text{ 次}}$$

迭代累次 $a^n = \underbrace{a^n (a^n \cdots (a^n 1) \cdots)}_{n \text{ 次}}$

—

单箭头 (single arrow), $a \uparrow n = a^n$

双箭头 (double arrow), $a \uparrow\uparrow n = \underbrace{(a \uparrow (a \uparrow \cdots (a \uparrow 1) \cdots)}_{n \text{ 次}}$ $= a^n$

三箭头 (triple arrow), $a \uparrow\uparrow\uparrow n = \underbrace{a \uparrow\uparrow (a \uparrow\uparrow \cdots (a \uparrow\uparrow 1) \cdots)}_{n \text{ 次}}$

已知三箭头的定义后可以递归地定义四箭头

即 $a \uparrow\uparrow\uparrow n = \underbrace{\overbrace{a^a \cdots a}^{a^a \cdots a} \cdots}_{n \text{ 层}}^a$

按照运算的定义: $\underbrace{a^a \cdots a}_{a^a \cdots a}$
n 层, $a \uparrow\uparrow\uparrow n = \underbrace{\overbrace{a^a \cdots a}^{a^a \cdots a} \cdots}_{n \text{ 层}}^a$

于是有四箭头 $\underbrace{a^a \cdots a}_{a^a \cdots a} \underbrace{a^a \cdots a}_{a^a \cdots a}$
 $n \text{ 次 } a \uparrow\uparrow\uparrow\uparrow n = \underbrace{\overbrace{a^a \cdots a}_{a^a \cdots a} \cdots}_{n \text{ 层}}^a$

$\underbrace{a^a \cdots a}_{a^a \cdots a} \underbrace{a^a \cdots a}_{a^a \cdots a}$
 $n \text{ 次 }$

特别地定义 \uparrow^n 为 $\underbrace{\uparrow \cdots \uparrow}_{n \text{ 层}}$ ($n \in \mathbb{Z}^+$), 并可定义 \uparrow^0 为乘法运算, 即 $a \uparrow^0 n = a * n$

又特别注意, 对于任意 $n \in \mathbb{N}$, $a \uparrow^n 1 = a$, 其中 $a \in \mathbb{Z}$

则有对于任意 $n \in \mathbb{Z}^+$, $a \uparrow^n b = \underbrace{a \uparrow^{n-1} (a \uparrow^{n-1} \cdots (a \uparrow^{n-1} a) \cdots)}_{b \geq 2}$

则有 $a \uparrow^n b = a \uparrow^{n-1} (a \uparrow^{n-1} \cdots (a \uparrow^{n-1} (a \uparrow^{n-1} 1)) \cdots) = (a \uparrow^{n-1})^b$

特别地有克努特箭头是一个右结合的运算符, 即 $a \uparrow b \uparrow c = a \uparrow (b \uparrow c)$

而乘法运算是合律的, 即也可以认为是右结合的, 所以扩展的 \uparrow^0 定义也是正确的

可以递归地定义 $a \uparrow^n b$ 运算, 其中 $a \in \mathbb{Z}$, $b, n \in \mathbb{N}$

$$a \uparrow^n b = \begin{cases} ab & , n=0 \\ 1 & , n \geq 1 \text{ 且 } b=0 \\ a^b & , n=1 \text{ 且 } b>0 \\ a \uparrow^{n-1} (a \uparrow^n (b-1)) & , n>1 \text{ 且 } b>0 \end{cases}$$

knuthArrow :: Int → Int → Int → Int

knuthArrow a 0 b = a * b

knuthArrow a n 0 = 1

knuthArrow a 1 b = a^b knuthArrow a n (b-1)

knuthArrow a n b = knuthArrow a (n-1)

考虑克努特箭头的单调性, 这里只考虑 $a \in \mathbb{Z}^+$ 的情形

当 $a=1$ 时, 当 $n=0$ 时, $a \uparrow^0 b = b$, 对于 b 是单调增加的 (严格地)

当 $n \geq 1$ 时, $a \uparrow^n b = 1$, 是单调增加 (非严格地, 因为全部相等), 对于 $b \neq 0$

当 $a>1$ 时, 当 $b=0$ 时, $a \uparrow^n b = 1$, 对于 n 是单调增加 (非严格, 全部相等)

当 $b>0$ 时, 对于任意 $n \in \mathbb{N}$, 都有 $a \uparrow^{n+1} b \geq a \uparrow^n b$, 且 $a \uparrow^n (b+1) \geq a \uparrow^n b$

前者在 $b=1$ 时取等号,

Discrete

Mathematics - P113

基于前述的阿克曼函数的定义，~~需要证明~~，~~通过数学归纳法~~，~~可以证明~~，~~对于任意~~ $m \in \mathbb{Z}^+$ ~~且~~ $n \in \mathbb{Z}^+$ ，~~有~~ $A(m, n) = 2 \uparrow^{n-1} m$

~~对于任意~~ $m \in \mathbb{Z}^+$ ~~且~~ $n \in \mathbb{Z}^+$ ，~~有~~ $A(m, n) = 2 \uparrow^{n-1} m$

~~基础步骤：~~当 $n=1$ 时， $A(m, 1) = 2m = 2 \uparrow^0 m$

当 $n=2$ 时， $A(m, 2) = 2^m = 2 \uparrow^1 m$

当 $n=3$ 时， $A(m, 3) = {}^m 2 = 2 \uparrow^2 m$

~~递归步骤：~~假设当 $m \in \mathbb{Z}^+$ 时，对任意 $n \in \mathbb{Z}^+$ 且 $n \geq 3$ ，有 $P_1 \wedge \dots \wedge P_n$ 为真，则考虑 P_{n+1}

$$A(m, n+1) = A(A(m-1, n+1), n) = A(A(A(m-2, n+1), n), n)$$

~~根据归内假设， $A(m, n) = 2 \uparrow^{n-1} m$~~ $\stackrel{\text{IH}}{=} 2 \uparrow^{n-1} (2 \uparrow^{n-1} (\dots 2 \uparrow^{n-1} (A(m-1, n+1)) \dots))$ ，~~有~~ $m-1$ 次 $2 \uparrow^{n-1}$ 运算

$$\text{又 } A(m-1, n+1) = 2 = 2 \uparrow^{n-1} 1, \quad \text{即 } m+n = 0 \Rightarrow 2 \uparrow^{n-1} 1$$

~~则有~~ $A(m, n+1) = 2 \uparrow^{n-1} (2 \uparrow^{n-1} (\dots 2 \uparrow^{n-1} (2 \uparrow^{n-1} 1) \dots))$ ，~~有~~ m 次 $2 \uparrow^{n-1}$ 运算

$$(2 \uparrow^{n-1})^m = (2 \uparrow^{n-1})^m 1 = 2 \uparrow^n m, \quad \text{即 } P_{n+1} \text{ 为真}$$

于是根据强归纳法，~~有~~ 对于任意 $m \in \mathbb{Z}^+, n \in \mathbb{Z}^+$ ， $A(m, n) = 2 \uparrow^{n-1} m$

对于任意非负整数 m 和 n ，~~有~~ $A(m, n) \geq A(m, n)$ ， $A(m+1, n) > A(m, n)$ ， $A(m, n) > m$

可以列表计算阿克曼函数 $A(m, n)$ 的值为

$n \backslash m$	0	1	2	3	4	\dots	m
0	$2 \uparrow^0 0 = 1$	2	4	5	6	\dots	$m+2$
1	$2 \uparrow^1 0 = 2$	$2 \downarrow 2 = 4$	6	8	16	\dots	$2m = 2 \uparrow^0 m$
2	$2 \uparrow^2 0 = 4$	$2 \uparrow^2 1 = 16$	$2 \uparrow^2 2 = 256$	$2 \uparrow^2 3 = 65536$	$2 \uparrow^2 4 = 2^{65536}$	\dots	$2^m = 2 \uparrow^1 m$
3	$2 \uparrow^3 0 = 8$	$2 \uparrow^3 1 = 2^{256}$	$2 \uparrow^3 2 = 2^{65536}$	$2 \uparrow^3 3 = 2^{2^{65536}}$	$2 \uparrow^3 4 = 2^{2^{2^{65536}}}$	\dots	${}^m 2 = 2 \uparrow^2 m$
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
n	$2 \uparrow^{n-1} 0$	$2 \uparrow^{n-1} 1$	$2 \uparrow^{n-1} 2$	$2 \uparrow^{n-1} 3$	$2 \uparrow^{n-1} 4$	\dots	$2 \uparrow^{n-1} m$

首先可以确定当 $n=0$ 时， $A(0, 0) = 1 > 0$ ， $A(1, 0) = 2 > 1$ ， $A(m, 0) = m+2 > m$

然后考虑当 $n=0$ 时， $A(m, 1) > A(1, 0) > A(0, 0)$

且如上所述，而对任意 $m \geq 2$ ， $A(m+1, 0) = m+3 > m+2 = A(m, 0)$

(当 $n=1$ 时， $A(1, 1) > A(0, 1)$)

且如上所述，而对任意 $m \geq 1$ ， $A(m+1, 1) \leq 2(m+1) > 2m = A(m, 1)$

且如上所述，当 $n \geq 2$ 时，对任意 $m \in \mathbb{N}$ ， $A(m+1, n) = 2 \uparrow^{n-1}(m+1) > 2 \uparrow^{n-1} m = A(m, n)$

最后考虑 当 $m=0$ 时， $A(0, 2) = A(0, 1) = A(0, 0)$ ，对任意 $n \geq 2$ ， $A(0, n) = 2 \uparrow^n 0 = 1 = A(0, n)$

当 $m=1$ 时， $A(1, 1) = A(1, 0)$ ，对任意 $n \geq 1$ ， $A(1, n+1) = 2 \uparrow^n 1 = 2 = 2 \uparrow^{n-1} 1 = A(1, n)$

当 $m \geq 2$ 时， $A(m, 1) = 2m \geq m+2 = A(m, 0)$ ，对任意 $n \geq 1$ ， $A(m, n+1) = 2 \uparrow^n m \geq 2 \uparrow^{n-1} m = A(m, n)$

$A(m, n+1) = 2 \uparrow^n m \geq 2 \uparrow^{n-1} m = A(m, n)$ ，仅当 $m=2$ 时，能取得等号

于是综合以上论证，有对任意 $m, n \in \mathbb{N}$ ， $A(m, n+1) \geq A(m, n)$ ， $A(m+1, n) > A(m, n)$ ， $A(m, n) > m$

Discrete

Mathematics - P114

超运算序列 (hyperoperation sequence) 是一个定义在整数上的二元运算的无限序列

前四项分别是加法, 乘法, 幂, 迭代幂次

序列中的每一项都是重复执行前一项运算, 注意这个定义与克努特箭头类似

如 乘法运算 $a \cdot n = \underbrace{a + a + \dots + a}_{n \text{次}}$, 重复了 n 次加法运算

超运算 (hyperoperation), 又称 hyper 运算符, 对应于超运算序列的一项

使用运算符 $[n]$, 表示序列中第 n 项运算, 称为超- n 运算或第 n 级超运算

超运算可以递归地定义, 对于任意正整数 a, b, n

基础步骤: $a[1]b = a + b$

递归步骤: 对于任意 $n > 1$, $a[n]b = \overbrace{a[n-1](a[n-1](a[n-1]\dots(a[n-1](a[n-1]a))\dots))}^b$

于是有对于 $n \geq 2$, 超- n 运算可以表示为 $n-2$ 克努特箭头运算

即对于 $n \geq 2$, $a[n]b = a \uparrow^{n-2} b$

由于 $a \cdot b = a + [a \times (b-1)]$ 表示乘法运算 \rightarrow 重复加法运算的步骤

于是可以用 $a + b = 1 + [a + (b-1)]$ 表示加法运算 \rightarrow 重复加法运算的步骤

于是可以扩展超运算序列 H_n 至定义在 $n \in \mathbb{N}$ 上, 即超-0 运算为后继运算

递归地定义运算 $H_n(a, b)$, 其中 $a, b, n \in \mathbb{N}$

hyperOperation :: Int → Int → Int → Int

hyperOperation _ 0 b = b+1

hyperOperation a 1 0 = a

hyperOperation a 2 0 = 0

hyperOperation a $\overset{n}{\uparrow} 0 = 1$

hyperOperation a n b = hyperOperation a (n-1) b

$$H_n(a, b) = \begin{cases} b+1, & n=0 \\ a & , n=1 \wedge b=0 \\ 0 & , n=2 \wedge b=0 \\ 1 & , n \geq 3 \wedge b=0 \\ H_{n-1}(a, H_n(a, b-1)), & a \geq 0 \\ & b > 0 \end{cases}$$

运算

定义

名称

定义域

克努特箭头

$$a[0]b = 1+b$$

$$1 + \underbrace{1 + 1 + \dots + 1}_b$$

hyper0, 后继函数

任意 b , arbitrary

—

$$a[1]b = a+b$$

$$a + \underbrace{1 + 1 + \dots + 1}_b$$

hyper1, 加法

arbitrary

—

$$a[2]b = ab$$

$$a \underbrace{\cdot a \cdot a \cdot \dots \cdot a}_b$$

hyper2, 乘法

arbitrary

$a \uparrow^0 b = ab$

$$a[3]b = a^b$$

$$1 \times a \times a \times \dots \times a$$

hyper3, 幂

$a > 0, b \in \mathbb{R}$ 或 $a \neq 0, b \in \mathbb{C}$ (扩展为复数)

$a \uparrow^1 b = a^b$

$$a[4]b = {}^b a$$

$$a \overset{b}{\wedge} a \overset{b}{\wedge} \dots \overset{b}{\wedge} a$$

hyper4, 迭代幂次

$a > 0, b \geq -1$ 且 $a, b \in \mathbb{Z}$

$a \uparrow^2 b = {}^b a$

$$a[5]b = a[4]a[4] \dots a[4]b$$

$$a[4] \overset{b}{\wedge} a[4] \dots a[4]b$$

hyper5, pentation

$a > 0, b \geq -1$ 且 $a, b \in \mathbb{Z}$

$a \uparrow^3 b$

$$a[6]b = a[5]a[5] \dots a[5]b$$

$$a[5] \overset{b}{\wedge} a[5] \dots a[5]b$$

hyper6, hexation

$a > 0, b \geq -1$ 且 $a, b \in \mathbb{Z}$

$a \uparrow^4 b$

$$a[n]b = a[n-1]a[n-1] \dots a[n-1]b$$

$$a[n-1] \overset{b}{\wedge} a[n-1] \dots a[n-1]b$$

hypern, $a = b$

$a > 0, b \geq -1$ 且 $a, b \in \mathbb{Z}$

$a \uparrow^{n-2} b$

特别地定义 $a[-1]b = b$, 定义域为 $a, b \in \mathbb{C}$

$n \geq 4$ 时, $a[n](-1) = 0$, 定义域为 $a, b \in \mathbb{C}$

Discrete

Mathematics - P115

STUDY

$p \rightarrow q = \text{transformation}$

康威链式箭号记法 (Conway chained arrow notation), 由 John Horton Conway 发明, 用于表示大整数。

递归地定义 Conway chain (康威链)

基础步骤: 任意正整数 p 是一个长度为 1 的链

递归步骤: 对于长度为 $n \in \mathbb{Z}^+$ 的链 X , 和正整数 p

加入右箭头表示的 $X \rightarrow p$ 是一个长度为 $n+1$ 的链

定义链的等价 (equivalent) 为, 如果两个链表示 (represent) 同一个整数

对于正整数 p 和 q , 和一个较短的康威链 X , 链所表示的值 $\boxed{\text{通过以下规则得到}}$

链 p 表示正整数 p . (长度为 1 的链)

链 $p \rightarrow q$ 表示正整数的幂 p^q (长度为 2 的链)

链 $X \rightarrow p \rightarrow 1$ 等价于 $X \rightarrow p$ $\boxed{X \rightarrow 1 \rightarrow (q+1)}$

链 $X \rightarrow p \rightarrow \overbrace{q+1}^{(q+1)}$ 等价于 $X \rightarrow (X \rightarrow (\cdots (X \rightarrow (X \rightarrow q) \cdots) \rightarrow q) \rightarrow q)$ $\boxed{\text{长度大于等于 } 3 \text{ 的链}}$

并且 $X \rightarrow p \rightarrow (q+1)$ 的展开可以用 $\boxed{\text{递归地定义}}$

基础步骤: $X \rightarrow 1 \rightarrow (q+1) = X$

递归步骤: $X \rightarrow (p+1) \rightarrow (q+1) = X \rightarrow (X \rightarrow p \rightarrow (q+1)) \rightarrow q$

注意所有的康威链都可以通过上述规则定义

长度为 3 的康威链可以等价地表示为超运算 (hyperoperation) 和克努特箭头 (Knuth's up-arrow)

即对于正整数 p, q, r , $p \rightarrow q \rightarrow r = p [r+2] q = p \uparrow^r q$

对于较短的康威链 X 和 Y , $X \rightarrow Y$ 形式上如同 $X \rightarrow p$, 其中 p 为正整数

a chain $X \rightarrow Y$ is of the form $X \rightarrow p$

以 a 开始的康威链所表示的正整数是 a 的幂

a chain starting with a is a power of a

链 $1 \rightarrow Y$ 等价于 1 , 即有 $1 \rightarrow Y = 1$

链 $X \rightarrow 1 \rightarrow Y$ 等价于 X , 即有 $X \rightarrow 1 \rightarrow Y = X$

$$= 2 \rightarrow 2 = 2^2 = 4$$

$$= 2 \rightarrow 2 \rightarrow (p-1) = \dots = 2 \rightarrow 2 \rightarrow 1$$

链 $2 \rightarrow 2 \rightarrow Y$ 等价于 4 , 即有 $2 \rightarrow 2 \rightarrow Y = 4$, 由于 $2 \rightarrow 2 \rightarrow p = 2 \rightarrow (2 \rightarrow 1 \rightarrow p) \rightarrow (p-1)$

链 $X \rightarrow 2 \rightarrow 2$ 等价于 $X \rightarrow (X)$, 其中 (X) 为链 X 所表示的正整数

如 $a \rightarrow b \rightarrow 2 \rightarrow 2 = a \rightarrow b \rightarrow (a \rightarrow b) = a \rightarrow b \rightarrow a^b = X \rightarrow (X \rightarrow 1 \rightarrow 2) \rightarrow 1$

$= X \rightarrow (X)$

特别注意, 康威链式箭号不是二元运算符, 也不符合交换律和结合律, 尤其是除非左结合也非右结合

如 $2 \rightarrow 3 \rightarrow 2 = 2 \uparrow^2 3 = {}^3 2 = 16 = 2^{2^2}$

$2 \rightarrow (3 \rightarrow 2) = 2 \rightarrow 9 = 2^9 = 512 = 2^{(3^2)}$

$(2 \rightarrow 3) \rightarrow 2 = 8 \rightarrow 2 = 8^2 = 64 = (2^3)^2$

Discrete

Mathematics - P116

递归算法 (recursive algorithm), 通过把问题递归到带有更小输入的相同问题是它的实例来解决原问题

递归摸指数 对于整数 $b \leq b < m$, $n \geq 0$, $m \geq 2$, 计算 $b^n \bmod m$ 的值

引入递归地定义 $\begin{cases} 1 & , n=0 \\ (b^{n/2} \bmod m)^2 \bmod m & , n>0 \text{ 且 } 2|n \\ [c b^{n/2}] \bmod m \cdot b \bmod m \bmod m & , n>0 \text{ 且 } 2 \nmid n \end{cases}$

procedure mpower (整数 b, n, m) | mpower :: Int → Int → Int → Int

if $n=0$ then
return 1

else if n 是偶数 then

return $(mpower(b, n/2, m))^2 \bmod m$

else

return $(mpower(b, \lfloor n/2 \rfloor, m))^2 \bmod m \cdot b \bmod m \bmod m$

注意递归算法的正确性可以通过数学归纳法和强归纳法证明

即证明算法对所有可能的输入值，都能产生正确的输出结果

迭代 (iteration), 基于反复利用循环中的操作的过程

从一个个或多个整数点处的函数值开始，连续地应用递归函数以在较大整数点处的函数值

1) 归并排序 (merge sort), 将一个表分成两个，并各自进行归并排序，并将排序结果归并成一个有序表

procedure merge (已排序的表 L₁, L₂) | merge :: [Int] → [Int] → [Int]

L := 空表

while L₁ 非空 and L₂ 非空 do

取 L₁ 和 L₂ 的首个元素，取其较小的一个

从原表中删除并添加到 L 的末尾

将 L₁, L₂ 添加到 L 之后

return L

procedure mergesort (L = a₁, a₂, ..., a_n)

if n > 1 then m = L[n/2]

L₁ = a₁, a₂, ..., a_m

L₂ = a_{m+1}, ..., a_n

L = merge (mergesort(L₁), mergesort(L₂))

return L

merge [] lst2 = lst2

merge lst1 [] = lst1

merge lst1 lst2 = if head lst1 < head lst2

then head (lst1) : (merge tail lst1) lst2

else head (lst2) : (merge lst1 (tail lst2))

mergeSort :: [Int] → [Int]

mergeSort [x] = [x]

mergeSort lst =

Case splitAt ((length lst) `div` 2) lst of

(lst1, lst2) → merge (mergeSort lst1) (mergeSort lst2)

合并 m 个元素与 n 个元素的有序表，使用不超过 m+n-1 次比较

归并排序的时间复杂度 T(n) = O(n log n)

Discrete Mathematics - P117

快速排序算法 (quick sort), 是一个有效的排序算法, 时间复杂度为 $T(n) = \Theta(n \log n)$

基础步骤: 当列表为空或者只有一个元素时, 直接返回列表作为结果

递归步骤: 取列表第 1 个元素 a_1 , 将列表 L 分成两份列表 $L_1 [1 2 3 4 5 6 7 8] L_2 [9]$

L_1 包含 L 中小于 a_1 的元素, L_2 包含 L 中大于 a_1 的元素

对 L_1 和 L_2 递归地应用快速排序算法,

返回 $L = L_1 + a_1 + L_2$

`quickSort :: [Int] → [Int]`

`quickSort [] = []`

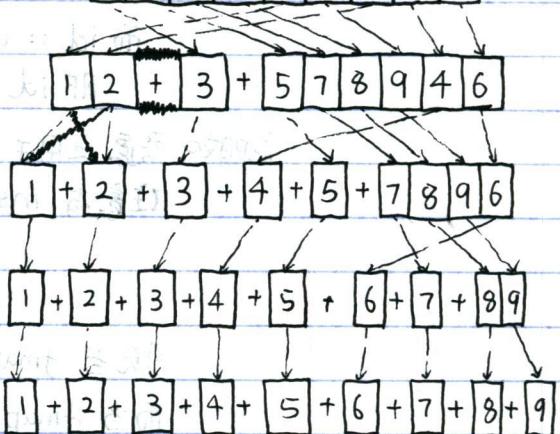
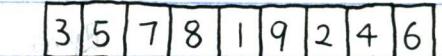
`quickSort [x] = [x]`

`quickSort lst = let a1 = head lst`

`lst1 = filter (\x → x < a1) (tail lst)`

`lst2 = filter (\x → x ≥ a1) (tail lst)`

`in (quickSort lst1) ++ [a1] ++ (quickSort lst2)`



程序验证 (program verification), 若对每个可能的输入, 程序都产生正确的输出, 则该程序是正确的

程序正确性 (program correctness), 指对过程总是产生正确结果的验证, 包含两个部分

如果程序终止, 则获得正确答案, 这部分称为程序的部分正确性 (partial correctness)

另一部分证明, 程序总是终止

使用两个命题来规定程序产生正确的输出, to prove output

初始断言 (initial assertion), 规定程序的输入值所具有的性质的命题

终结断言 (final assertion), 规定如果程序正确地工作则输出值所应当具有的性质的命题

霍尔逻辑 (Hoare Logic), 用于使用严格的数理逻辑推理来得到程序正确性的逻辑规则

对于一个程序或程序段 S , 如果对输入值有初始断言 P 为真时

就有对经过程序 S 的输出值有终结断言 Q 为真, 则说 S 是相对于 P 和 Q 是部分正确的

记为 $P \{S\} Q$, 称为霍尔三元组 (Hoare triple), 也有写作 $\{P\} S \{Q\}$.

注意如果程序 S 不终止, 则不存在对 Q 的真值判断, 所以 Q 根本上可以是任何语句

所以可以定义 Q 为 False 为程序 S 不终止, 则此时证明的是部分正确性

如果单独地证明程序 S 可能终止且终止时 Q 为真, 则称为完全正确性 (total correctness)

霍尔同时定义了关于 run-time error

failure to terminate may be due to an infinite loop,

or it may be due to violation of an implementation-defined limit

time limit

for example, the range of numeric operands, the size of storage, or an operating system

Discrete

Mathematics - P118

空语句公理 (empty statement axiom schema), 指如果命题 P 在空语句前成立, 则执行空语句后 P 也是成立的。记为 $\{P\} \text{skip} \{P\}$, 其中 skip 表示空语句 (empty statement)。

赋值公理 (assignment axiom schema), 指对赋值右端的变量为真的任何命题, 在赋值后仍然成立。
any predicate that was previously true for the right-hand side of the assignment
now holds for the variable after the assignment

对于包含自由变量 x 的断言 P , 记为 $\{P[E/x]\} x := E \{P\}$

其中 $P[E/x]$ 表示在断言 P 中自由变量 x 均用表达式 E 替换:

the assertion P in which each free occurrence of x has been replaced by the expression E

注意 $P[E/x]$ 的真值与赋值语句执行后的 P 的真值 等价

根据公理, $P[E/x]$ 在赋值前为真, 则 P 在赋值后为真

另外, 如果 $P[E/x]$ 在赋值前为假, 则在赋值后 P 必须为假

如 $\{x+1 = 43\} y := x+1 \{y = 43\}$

由于指令 $y := x+1$ 并未改变 $x+1 = 43$, 所以在 $x+1 = 43$ 都会出现在后置条件 (postcondition)

前置条件 (precondition): $P[(x+1)/y]$ 有 $\{x+1 = 43\}$ 为真

后置条件 $\{y = 43 \wedge x+1 = 43\}$ 中用 $x+1$ 替换 y 后, 有 $\{x+1 = 43 \wedge x+1 = 43\}$, 即 $\{x+1 = 43\}$ 为真

所以若取前置条件 (precondition) 的前提是, 对于后置条件 (post condition)

对于所有自由出现的赋值式左端的部分替换成赋值式右端的部分

但是注意反向执行, 如 $\{P\} x := E \{P[E/x]\}$ 是错误的

如 $\{x=5\} x := 3 \{3=5\}$ 是不合理的

另外注意如 $\{P\} x := E \{P \wedge x = E\}$ 看似正确, 但也会产生错误

如 $\{x=5\} x := x+1 \{x=5 \wedge x=x+1\}$

这是由于赋值式并非等式, 所以不能直接将赋值式改写为等式

注意对于给定的后置条件 P 唯一地有前置条件 $P[E/x]$, 但是反向过程不是唯一的

如给定前置条件 $\{0 \leq y \cdot y \wedge y \cdot y \leq 9\}$, 和赋值式 $x := y \cdot y$

$\{0 \leq x \wedge x \leq 9\}, \{0 \leq x \wedge y \cdot y \leq 9\}, \{0 \leq y \cdot y \wedge x \leq 9\}, \{0 \leq y \cdot y \wedge y \cdot y \leq 9\}$ 都是合法的后置条件

特别注意: 当命题中有不止一个变量名指向同一个存储值时,

应用霍尔变量会产生错误, 所以需要谨慎处理

如 $\{y=3\} x := 2 \{y=3\}$

当 x 和 y 引用同一个变量值时, 这个命题是假的

这种情形称为别名 (aliasing), 即内存中的一个数据位置, 可以通过程序的多个符号来访问

如果 x 与 y 是不同的变量, 在 $\{y=3\} x := 2 \{y=3\}$ 中

后置条件 $\{P\}$ 和前置条件 $\{P[2/x]\}$ 都是 $\{y=3\}$

Discrete

Mathematics - P119

合成规则 (rule of composition), 应用霍尔逻辑规则在顺序执行程序 (sequentially executed program)

对于顺序执行程序 $S \text{ 和 } T$, 如果 S 在 T 之前执行, 则写作 $S; T$

则对于初始断言 $\{P\}$ 和终结断言 $\{R\}$, 还有中间状态断言 (midcondition) $\{Q\}$

则记为 $\{P\} S \{Q\}; \{Q\} T \{R\}$ 或者写作 $\frac{P\{S_1\} Q}{Q\{S_2\} R}$

即假设已经证明了 $\frac{P\{S_1\} Q}{Q\{S_2\} R}$ 相对于初始断言 $\{P\}$ 和终结断言 $\{R\}$ 的正确性
以及 $\frac{Q\{S_2\} R}{P\{S_1\} Q}$ 相对于初始断言 $\{P\}$ 和终结断言 $\{R\}$ 的正确性

则有如果 P 为真且程序 $U = S; T$ 执行并终止, 则有 $\{R\}$ 为真

如 $\{x+1=43\} y := x+1 \{y=43\}$, 表示给定 $x+1=43$, 则在赋值语句 $y := x+1$ 后, 有 $y=43$

$\{y=43\} z := y \{z=43\}$, 表示给定 $y=43$, 则在赋值语句 $z := y$ 后, 有 $z=43$

即 $\{x+1=43\} y := x+1; z := y \{z=43\}$

表示给定 $x+1=43$, 则在语句 $y := x+1; z := y$ 执行后, 有 $z=43$ 为真

特别地有, $\{P\} S \{Q\}; \{Q\} T \{R\}$ 可以证明 $\{P\} S; T \{R\}$ 的正确性,

但是注意反向是不一定, 因为当 $\{P\} S; T \{R\}$ 为真时

可能存在一个不同于 $\{Q\}$ 的断言 $\{Q'\}$, 使得 $\{P\} S \{Q'\}; \{Q'\} T \{R\}$ 为真

条件规则 (conditional rule), 应用于条件语句的霍尔逻辑规则, 即形如 if B then S [else T] endif 的语句

对于一个语句块 S , 和一个条件断言 (condition) $\{B\}$, 初始断言 $\{P\}$ 和终结断言 $\{Q\}$

即记为 $\{B \wedge P\} S \{Q\}$

指仅有对条件断言为真时执行语句的情况

$\{\neg B \wedge P\} \rightarrow \{Q\}$

即 if $\{B\}$ then S endif

$\{P\} \text{ if } \{B\} \text{ then } S \{Q\}$

注意在这种情况下, 条件断言为假时的判断 并不为真

标准情形为 $\{B \wedge P\} S \{Q\}$

即 if B

对于不同的 $\{\neg B \wedge P\} T \{Q\}$

then S

语句块 S 和 T $\{P\} \text{ if } \{B\} \text{ then } S \text{ else } T \{Q\}$

else T endif

注意后置条件 $\{Q\}$ 也是 then 和 else 部分的后置条件,

加入前置条件

而在生成前置条件时需要将 unnegated condition B 和 negated condition $\neg B$

特别注意 B 本身不能有副作用 (side effect)

注意还可以扩展为 $\{C_1 \wedge P\} S_1 \{Q\}$

即表示为 switch: case $C_1 \rightarrow S_1$,

$\{C_2 \wedge P\} S_2 \{Q\}$

case $C_2 \rightarrow S_2$

:

$\{C_n \wedge P\} S_n \{Q\}$

case $C_n \rightarrow S_n$

$\{P\} \text{ case } \{C_i\} \text{ then } S_i \{Q\}$

其中 C_1, C_2, \dots, C_n 为条件表达式, S_1, S_2, \dots, S_n 是语句块

其中 $i = 1, 2, \dots, n$

注意条件断言应满足 $C_1 \vee C_2 \vee \dots \vee C_n \equiv T$