

§1 程序的推理验证

要证明一个程序或程序模型是否具备某些性质，我们可以用推理的方法。我们先介绍一个一般性的推理方法，再介绍一些特殊类型程序的推理方法。

§1.1 一阶迁移系统的推理

这里介绍以一阶迁移系统为程序模型、一阶线性时序逻辑为程序性质的语言的推理方法。设 (T, Θ) 为 B 上的一阶迁移系统。给定 B 上的解释 I 。 (T, Θ) 满足一阶线性时序逻辑公式 φ ，记作 $(T, \Theta) \models_I \varphi$ 。其定义如下：

$$\frac{(T, \Theta) \models_I \varphi}{\text{当且仅当}} \\ \frac{\text{对所有 } I \text{ 解释下的 } (T, \Theta) \text{ 的运行 } \pi, \pi \models_I \varphi}{(T, \Theta) \models_I \varphi}$$

给定 B 上的一阶迁移系统 (T, Θ) 。对 $t \in T, \varphi, \psi \in QFF_B$ ，一个迁移公式为三元组：

$$\{\varphi\}t\{\psi\}$$

给定 B 上的解释 I 。给定一个迁移 $t : p \rightarrow (x_1, \dots, x_n) = (e_1, \dots, e_n)$ ，和两个状态 σ, σ' 。定义 $\sigma \xrightarrow{t} \sigma'$ 当前仅当 $\sigma \models_I p$ 且 $\sigma' = \sigma[x_1/I(e_1)\sigma] \cdots [x_n/I(e_n)\sigma]$ 。一个迁移公式在 I 下是一个函数

$$I(\{\varphi\}t\{\psi\}) : \Sigma \rightarrow Bool$$

其定义如下：

$$\frac{I(\{\varphi\}t\{\psi\})(\sigma) = true}{\text{当且仅当}} \\ \frac{I(\varphi)(\sigma) = true \wedge \sigma \xrightarrow{t} \sigma' \rightarrow I(\psi)(\sigma') = true}{I(\{\varphi\}t\{\psi\})(\sigma) = true}$$

公式 $\{\varphi\}t\{\psi\}$ 在解释 I 下成立，当且仅当 $\forall \sigma \in \Sigma, I(\{\varphi\}t\{\psi\})(\sigma) = true$ ，记作

$$\models_I \{\varphi\}t\{\psi\}$$

给定 (T, Θ) 。我们有以下公理和规则：

Θ 规则：

$$\frac{\Theta \Rightarrow \varphi}{\varphi}$$

迁移公理：

$$\{p \rightarrow \psi[x/e_1] \cdots [x/e_n]\} p \longrightarrow (x_1, \dots, x_n) := (e_1, \dots, e_n) \{\psi\}$$

给定一个迁移 $t : p \rightarrow (x_1, \dots, x_n) = (e_1, \dots, e_n)$ 。定义 $COND_t = p$ 。给定一个迁移系统 (T, Θ) 。定义 $COND_T = \bigvee_{t \in T} COND_t$ 。

O 规则:

$$\frac{\varphi \Rightarrow (COND_T \vee \psi) \quad \forall t \in T, \{\varphi\}t\{\psi\}}{\varphi \Rightarrow O\psi}$$

推断规则:

$$\frac{\varphi' \Rightarrow \varphi \quad \{\varphi\}t\{\psi\} \quad \psi \Rightarrow \psi'}{\{\varphi'\}t\{\psi'\}}$$

由这些规则和一阶线性时序逻辑的推理可以导出以下规则:

迁移:

$$\frac{p \wedge \varphi \rightarrow \psi[x/e_1] \cdots [x/e_n]}{\{\varphi\}p \longrightarrow (x_1, \dots, x_n) := (e_1, \dots, e_n) \{\psi\}}$$

O 规则:

$$\frac{\varphi \Rightarrow COND_T \quad \forall t \in T, \{\varphi\}t\{\psi\}}{\varphi \Rightarrow O\psi}$$

U 性质

设 $w \in QFF_B$ 为一元谓词公式 (记其变量为 x); $e \in T_B$ 为项; $\sqsubseteq \in P$ 为二元谓词符号; $(\{\sigma(x) \mid I(w)(\sigma) = true\}, I_0(\sqsubseteq))$ 为良基集合。

U 规则:

$$\frac{\varphi \Rightarrow (\psi \vee \zeta) \quad \zeta \Rightarrow (\zeta_0 \wedge w_x^e \wedge (\psi \vee COND_T)) \quad \forall t \in T, \{\zeta \wedge e = v\}t\{\psi \vee (\zeta \wedge e \sqsubseteq v)\}}{\varphi \Rightarrow \zeta_0 U \psi}$$

◇ 规则:

$$\frac{\varphi \Rightarrow (\psi \vee \zeta) \quad \zeta \Rightarrow (w_x^e \wedge (\psi \vee COND_T)) \quad \forall t \in T, \{\zeta \wedge e = v\}t\{\psi \vee (\zeta \wedge e \sqsubseteq v)\}}{\varphi \Rightarrow \diamond \psi}$$

R 性质

R 规则:

$$\frac{\zeta \Rightarrow \varphi \quad \forall t \in T, \{\varphi \wedge \neg \psi\}t\{\varphi\} \quad \varphi \Rightarrow \varphi'}{\zeta \Rightarrow \psi R \varphi'}$$

□ 规则:

$$\frac{\zeta \Rightarrow \varphi \quad \forall t \in T, \{\varphi\}t\{\varphi\} \quad \varphi \Rightarrow \varphi'}{\zeta \Rightarrow \square \varphi'}$$

给定一个迁移 $t : p \rightarrow (x_1, \dots, x_n) = (e_1, \dots, e_n)$ 。定义 $EXPL_t = (e_1, \dots, e_n)$ 。EFLTL 变量规则:

$$\frac{\varphi \Rightarrow COND_T}{\varphi \Rightarrow \bigvee_{t \in T} (COND_t \wedge O\bar{x} = EXPL_t)} \quad \frac{\varphi \Rightarrow \neg COND_T}{\varphi \Rightarrow O\bar{x} = \bar{x}}$$

例子

B 上的迁移系统 (T, Θ)

T 为以下迁移:

$$\begin{array}{ll} a = s_0 & \rightarrow (y, t, a) := (1, 1, s_1) \\ a = s_1 \wedge (x = 0 \vee t = 0) & \rightarrow (a) := (s_2) \\ a = s_2 & \rightarrow (y, a) := (0, s_3) \\ a = s_3 & \rightarrow (y, t, a) := (1, 1, s_1) \\ b = t_0 & \rightarrow (x, t, b) := (1, 0, t_1) \\ b = t_1 \wedge (y = 0 \vee t = 1) & \rightarrow (b) := (t_2) \\ b = t_2 & \rightarrow (x, b) := (0, t_3) \\ b = t_3 & \rightarrow (x, t, b) := (1, 0, t_1) \end{array}$$

$$\Theta = (a = s_0 \wedge b = t_0 \wedge x = 0 \wedge y = 0 \wedge t = 0)。$$

证明 $\square(a = s_1 \wedge b \neq t_1 \wedge b \neq t_2 \rightarrow a = s_2 Rb \neq t_2)$ 。

尝试

$$\begin{array}{ll} \zeta & = (a = s_1 \wedge b \neq t_1 \wedge b \neq t_2) \\ \varphi & = (b \neq t_2) \\ \psi & = (a = s_2) \end{array}$$

需要 $\zeta \Rightarrow \varphi$ 和 $\forall t \in T, \{\varphi \wedge \neg\psi\}t\{\varphi\}$ 。

试证:

$$\{\varphi \wedge \neg\psi\}t_6\{\varphi\}。$$

需要

$$(b \neq t_2) \wedge (a \neq s_2) \wedge b = t_1 \wedge (y = 0 \vee t = 1) \rightarrow (t_2 \neq t_2)$$

令

$$\begin{array}{ll} \zeta & = (a = s_1 \wedge b \neq t_1 \wedge b \neq t_2) \\ \varphi & = (a = s_1 \wedge (b = t_0 \vee b = t_3 \vee (b = t_1 \wedge x = 1 \wedge t = 0))) \vee (a = s_2 \wedge b \neq t_2) \\ \psi & = (a = s_2) \\ \varphi' & = (b \neq t_2) \end{array}$$

试证:

$$\{\varphi \wedge \neg\psi\}t_6\{\varphi\}。$$

需要

$$\begin{aligned} & ((a = s_1 \wedge (b = t_0 \vee b = t_3 \vee (b = t_1 \wedge x = 1 \wedge t = 0))) \vee (a = s_2 \wedge b \neq t_2)) \wedge \\ & (b \neq t_2) \wedge (a \neq s_2) \wedge b = t_1 \wedge (y = 0 \vee t = 1) \\ & \rightarrow (a = s_1 \wedge (t_2 = t_0 \vee t_2 = t_3 \vee (t_2 = t_1 \wedge x = 1 \wedge t = 0))) \vee (a = s_2 \wedge t_2 \neq t_2) \end{aligned}$$

$$\begin{aligned}
& ((a = s_1 \wedge (b = t_0 \vee b = t_3 \vee (b = t_1 \wedge x = 1 \wedge t = 0))) \wedge \\
& (b \neq t_2) \wedge (a \neq s_2) \wedge b = t_1 \wedge (y = 0 \vee t = 1)) \\
& \rightarrow (a = s_1 \wedge (t_2 = t_0 \vee t_2 = t_3 \vee (t_2 = t_1 \wedge x = 1 \wedge t = 0))) \vee (a = s_2 \wedge t_2 \neq t_2) \\
& ((a = s_1 \wedge ((b = t_1 \wedge x = 1 \wedge t = 0))) \wedge (y = 0)) \\
& \rightarrow (a = s_1 \wedge (t_2 = t_0 \vee t_2 = t_3 \vee (t_2 = t_1 \wedge x = 1 \wedge t = 0))) \vee (a = s_2 \wedge t_2 \neq t_2) \\
& \quad \text{令}
\end{aligned}$$

$$\begin{aligned}
\zeta &= (a = s_1 \wedge b \neq t_1 \wedge b \neq t_2) \\
\varphi &= (a = s_1 \wedge (b = t_0 \vee b = t_3 \vee (b = t_1 \wedge x = 1 \wedge t = 0 \wedge y = 1))) \vee (a = s_2 \wedge b \neq t_2) \\
\psi &= (a = s_2) \\
\varphi' &= (b \neq t_2)
\end{aligned}$$

试证:

$$\{\varphi \wedge \neg\psi\} t_6 \{\varphi\}.$$

需要

$$\begin{aligned}
& ((a = s_1 \wedge (b = t_0 \vee b = t_3 \vee (b = t_1 \wedge x = 1 \wedge t = 0 \wedge y = 1))) \vee (a = s_2 \wedge b \neq t_2)) \wedge \\
& (b \neq t_2) \wedge (a \neq s_2) \wedge b = t_1 \wedge (y = 0 \vee t = 1) \\
& \rightarrow (a = s_1 \wedge (t_2 = t_0 \vee t_2 = t_3 \vee (t_2 = t_1 \wedge x = 1 \wedge t = 0 \wedge y = 1))) \vee (a = s_2 \wedge t_2 \neq t_2) \\
& ((a = s_1 \wedge (b = t_0 \vee b = t_3 \vee (b = t_1 \wedge x = 1 \wedge t = 0 \wedge y = 1))) \wedge \\
& (b \neq t_2) \wedge (a \neq s_2) \wedge b = t_1 \wedge (y = 0 \vee t = 1)) \\
& \rightarrow (a = s_1 \wedge (t_2 = t_0 \vee t_2 = t_3 \vee (t_2 = t_1 \wedge x = 1 \wedge t = 0 \wedge y = 1))) \vee (a = s_2 \wedge t_2 \neq t_2)
\end{aligned}$$

因此 $a = s_1 \wedge b \neq t_1 \wedge b \neq t_2 \Rightarrow a = s_2 R b \neq t_2$,

即 $\Box(a = s_1 \wedge b \neq t_1 \wedge b \neq t_2 \rightarrow a = s_2 R b \neq t_2)$ 。

分两种情况来证明, 分析的时候可能方便些。

我们有 $\zeta \rightarrow \zeta_1 \vee \zeta_2$,

其中 $\zeta_1 = (a = s_1 \wedge b = t_0)$, $\zeta_2 = (a = s_1 \wedge b = t_3)$,

令

$$\varphi_1 = (a = s_1 \wedge (b = t_0 \vee (b = t_1 \wedge x = 1 \wedge t = 0 \wedge y = 1))) \vee (a = s_2 \wedge b \neq t_2)$$

则 $\zeta_1 \Rightarrow \psi R \varphi_1$ 。

因此 $\zeta_1 \Rightarrow a = s_2 R b \neq t_2$ 。

同理 $\zeta_2 \Rightarrow a = s_2 R b \neq t_2$ 。

因此 $\zeta \Rightarrow a = s_2 R b \neq t_2$ 。

...

例子

证明 $\Box(a = s_1 \rightarrow \Diamond a = s_2)$ 。

选择 f 满足

$$\begin{aligned}
I(f(t_0, 0)) &= 1 & I(f(t_1, 0)) &= 0 & I(f(t_2, 0)) &= 2 & I(f(t_3, 0)) &= 1 \\
I(f(t_0, 1)) &= 1 & I(f(t_1, 1)) &= 3 & I(f(t_2, 1)) &= 2 & I(f(t_3, 1)) &= 1
\end{aligned}$$

令

$$\begin{aligned}
W &= (\{0, 1, 2, 3\}, \leq) \\
w &= (0 \leq x \leq 3) \\
e &= f(b, t) \\
\varphi &= (a = s_1) \\
\psi &= (a = s_2) \\
\zeta &= (a = s_1)
\end{aligned}$$

需要

$$\begin{aligned}\varphi &\Rightarrow (\psi \vee \zeta) \\ \zeta &\Rightarrow w_x^e \wedge (\psi \vee \bigvee_{t \in T} COND_t) \\ \forall t \in T, \{\zeta \wedge e = v\} &t\{\psi \vee (\zeta \wedge e < v)\}\end{aligned}$$

试证:

$$\{\zeta \wedge e = v\} t_6 \{\psi \vee (\zeta \wedge e < v)\}.$$

需要

$$\begin{aligned}((a = s_1 \wedge f(b, t) = v) \wedge b = t_1 \wedge (y = 0 \vee t = 1)) \\ \rightarrow (a = s_2 \vee (a = s_1 \wedge f(t_2, t) < v))\end{aligned}$$

$$\begin{aligned}((a = s_1 \wedge f(t_1, t) = v) \wedge b = t_1 \wedge (y = 0 \vee t = 1)) \\ \rightarrow (a = s_2 \vee (a = s_1 \wedge f(t_2, t) < v))\end{aligned}$$

令

$$\begin{aligned}W &= (\{0, 1, 2, 3\}, \leq) \\ w &= (0 \leq x \leq 3) \\ e &= f(b, t) \\ \varphi &= (a = s_1) \\ \psi &= (a = s_2) \\ \zeta &= (a = s_1 \wedge y = 1)\end{aligned}$$

因此 $\varphi \Rightarrow \diamond\psi$ 。

习题

设 $B = (\{x, y, n, a\}, \{s_0, s_1, s_2, s_3, s_4, 0, 1, 2, 3, \dots, +, -, *\}, \{<, =, >\})$ 。给定 B 上的迁移系统 (T, Θ) ，其中 Θ 为 $a = s_0$ 且 T 为以下迁移:

$$\begin{aligned}a = s_0 &\longrightarrow (x, y, a) := (0, 0, s_1) \\ a = s_1 \wedge x < n &\longrightarrow (a) := (s_2) \\ a = s_2 &\longrightarrow (y, x, a) := (y + x * (x + 1), x + 1, s_1) \\ a = s_1 \wedge \neg(x < n) &\longrightarrow (a) := (s_3) \\ a = s_3 &\longrightarrow (y, a) := (3 * y, s_4)\end{aligned}$$

给定 I 为 B 在整数上的正常解释。证明以下命题成立:

$$\begin{aligned}(1) \quad (T, \Theta) \vdash_I n \geq 0 \rightarrow \Box(a = s_4 \rightarrow y = n * n * n - n) \\ (2) \quad (T, \Theta) \vdash_I n \geq 0 \rightarrow \Diamond(a = s_4)\end{aligned}$$

§1.1.1 推理系统的可靠与完备

推理系统的可靠与完备

可靠性是指只有正确的公式能够由推理系统的公理和规则推导得出，即

$$\vdash_I \varphi$$

则

$$\models_I \varphi$$

系统可靠性的证明一般有两个方面就是所有由公理产生的公式都是正确的且对于所有推导规则，只要前提正确，则结论正确。

完备性是指所有正确的公式能够由推理系统的公理和规则推导得出，即

$$\models_I \varphi$$

则

$$\vdash_I \varphi$$

一个不完备的推理系统可以用来证明某些正确的公式，但其使用有很大局限性的。程序的推理系统由两个方面：一个是有关程序语句或迁移的推理，另一个是程序所依赖的基础逻辑的推理。由于逻辑的复杂性，某些逻辑是不可能完备的推理系统的。比如自然数域上的一阶逻辑是不可能完备的推理系统的。又由于逻辑描述的语法限制，推理过程中需要用到的某些性质可能没法表达。比如给定一个一阶逻辑，一个二元组是否属于一个二元谓词的传递闭包是这个逻辑没法表达的。基于以上原因程序的推理系统很难有完备的。为了能够分清与程序相关的推理和其所依赖的基础逻辑的推理，我们将一个程序推理的不同组成部分分开，引入相对完备性的概念。一个程序推理系统是相对完备的，即在下列假设的前提下，系统是完备的：

基础逻辑的正确性质都作为程序推理系统的公理。

程序推理过程中需要用到的基础逻辑的性质都有相应的公式可以表达。

基于一阶线性时序逻辑的程序推理系统的相对完备性的证明，可以转换成程序性质的验证条件（用时序逻辑公式表达）可由程序有关的推理规则得到的证明。首先，我们可以用时序逻辑刻画 (T, Θ) 。

$$\Phi = \Theta \wedge \Box(\text{COND}_T \rightarrow \bigvee_t (\text{COND}_t \wedge O\bar{x} = \text{EXPL}_t)) \wedge \Box(\neg \text{COND}_T \rightarrow O\bar{x} = \bar{x})$$

显然，这个公式是可以由程序推理规则推导出的，即

$$(T, \Theta) \vdash_I \Phi$$

剩下的就是证明 $(T, \Theta) \models \varphi$ 则 $\vdash_I \Phi \rightarrow \varphi$ 。由于 Φ 完全刻画了 (T, Θ) 的初始条件和状态转换关系，可以根据语义证明这是成立的。

§1.2 流程图程序的推理

流程图程序是一阶逻辑的扩充。

有语法规则和语义描述。

流程图

给定符号集 F 和 P ，我们可以在 $B = (V, F, P)$ 上建立一个一阶逻辑。

在此基础上，我们增加以下集合的符号：

$$\begin{aligned} AUX &= \{:=, ;, :, goto, if, else\} \\ LAB &= \{begin, end, l_1, l_2, \dots\} \end{aligned}$$

B 上的流程图程序的指令有两种形式。定义如下：

- $l_1 : (x_1, \dots, x_n) := (t_1, \dots, t_n); goto l_2$
其中 $l_1, l_2 \in LAB$,
 $l_1 \neq end$,
 $t_1, \dots, t_n \in T_B$,
 $x_1, \dots, x_n \in V$
且对 $0 \leq i < j \leq n, x_i \neq x_j$ 。
- $l_1: if (e) goto l_2 else goto l_3$
其中
 $l_1, l_2, l_3 \in LAB$,
 $l_1 \neq end, l_2 \neq l_3$,
 $e \in QFF_B$.

一个指令中冒号左边的标号被冒号右边的式子所定义。 B 上的流程图程序 T 为 B 上指令的集合，且 T 满足以下条件。

- 标号的定义不重复。
- 所有出现在 T 中的标号除了 end 都有定义。
- 标号 beg 需在 T 中出现。

B 上流程图程序的集合记为 \mathcal{L}_1^B 。

例子

设

$$B' = (\{x, y_1, y_2, y_3\}, \{0, 1, 2, 3, \dots, +, *\}, \{\leq\})$$

以下程序，记为 T_0 ，是 \mathcal{L}_1^B 的一个例子：

```
beg:   (y1, y2, y3) := (0, 1, 1); goto test
test:  if (y3 ≤ x) goto loop else goto end
loop:  (y1, y2) := (y1 + 1, y2 + 2); goto inloop
inloop: y3 := y3 + y2; goto test
```

运行

给定 $B = (V, F, P)$ 的一个解释 I 。

我们将其扩充到 $B' = (V \cup \{lab\}, F \cup LAB, P)$ 上的一个解释。

状态分 σ' 成两部分，

一部分为 lab 的值，

另一部分为其它变量的值，

记为 (l, σ) ，即 $\sigma'(lab) = l$ 且 $\sigma'(x) = \sigma(x)$

其中 l 为 $D_0 = \{I(a) \mid a \in LAB\}$ 的元素, 用以代表 lab 的值。

程序 T 的状态转换关系记为 $\overset{T}{\Rightarrow}$ 。

$(l_i, \sigma_i) \overset{T}{\Rightarrow} (l_{i+1}, \sigma_{i+1})$ 当且仅当以下一项成立。

- 存在指令 $l_i: (v_1, v_2, \dots, v_n) := (e_1, e_2, \dots, e_n) \text{ goto } l_{i+1}$
且 $\sigma_{i+1} = \sigma_i[v_1/I(e_1)(\sigma_i)] \cdots [v_n/I(e_n)(\sigma_i)]$
- 存在指令 $l_i: \text{if } (b) \text{ goto } l' \text{ else goto } l''$, $\sigma_{i+1} = \sigma_i$ 且 $\sigma_i \models_I b$ 则 $l_{i+1} = l'$, 否则 $l_{i+1} = l''$
- 不存在定义 l_i 的指令, 且 $l_i = l_{i+1}$ 且 $\sigma_i = \sigma_{i+1}$ 。

在不引起混淆的时候, 我们将 $\overset{T}{\Rightarrow}$ 中的 T 省略写成 \Rightarrow 。定义 $\overset{*}{\Rightarrow}$ 为 \Rightarrow 的自反闭包。 $\{(l, \sigma) \mid (l_0, \sigma_0) \overset{*}{\Rightarrow} (l, \sigma)\}$ 即是由 (l_0, σ_0) 可达的状态。

程序在初始变量状态为 σ 时的运行终止, 当且仅当存在 σ' 使得

$$(beg, \sigma) \overset{*}{\Rightarrow} (end, \sigma')$$

程序的语义函数是从状态到状态的函数 $\mathcal{M}_I(T) : \Sigma \mapsto \Sigma$, 定义如下。

$$\mathcal{M}_I(T)(\sigma) = \begin{cases} \sigma' & \text{若 } (beg, \sigma) \overset{*}{\Rightarrow} (end, \sigma')。 \\ \text{无定义} & \text{若程序在初始变量状态为 } \sigma \text{ 时的运行不终止。} \end{cases}$$

$\mathcal{M}_I(T)(\sigma)$ 有定义

或者说程序 T 在初始变量状态为 σ 时的运行终止

记作 $\mathcal{M}_I(T)(\sigma) \downarrow$ 。

流程图程序可与一阶迁移系统

流程图程序可以看成是一阶迁移系统的子类。

一个一阶迁移系统有一个基本部分, 就是基本符号集

$$B = (V, F, P)$$

我们将 V 分为两部分

$$V = \{lab\} \cup V_1$$

将 F 分为两部分

$$F = F_0 \cup F_1$$

F_0 和 F_1 无交集, 且 $beg, end \in F_0$ 。

设 $B' = (V_1, F_1, P)$ 。流程图程序是满足以下限制的 $B = (V, F, P)$ 上的一阶迁移系统 (T, Θ) 。

- 迁移的形式如下

$$lab = l \rightarrow (v_1, v_2, \dots, v_n, lab) := (e_1, e_2, \dots, e_n, l')$$

其中

$$l, l' \in F_0$$

$$v_1, v_2, \dots, v_n \in V_1$$

$$e_1, e_2, \dots, e_n \in T_{B'}$$

$$lab = l \wedge e \rightarrow (lab) := (l')$$

其中

$$l, l' \in F_0$$

$$e \in WFF_{B'}$$

• 迁移集合 T 的限制如下

- $lab = beg$ 是 T 的某个迁移的前提条件。
- 若 $lab = l$ 是 T 的某个迁移的前提条件，
则 $lab = l$ 不是 T 的其它迁移的前提条件。
- 若 $lab = l \wedge e$ 是 T 的某个迁移的前提条件，
则 $lab = l \wedge \neg e$ 是 T 的某个迁移的前提条件，
 $lab = l$ 不为 T 的其它迁移的前提条件
且不存在 $e' \notin \{e, \neg e\}$ 使得 $lab = l \wedge e'$ 为 T 的其它迁移的前提条件。
- end 不出现在 T 的前提条件中。
除 end 外的 F_0 中的元素，若其在 T 出现，
则其亦在 T 的某个迁移的前提条件中。

• 初始条件的形式为 $lab = beg$ 。

对于 B 的解释 $I = (D, I_0)$ ，
我们要求 $a, b \in F_0, a \neq b$ 则 $I_0(a) \neq I_0(b)$ 。
这样，我们能够保证系统运行的确定性。

§1.2.1 程序性质和证明

对于流程图程序，传统的性质包括部分正确和完全正确。

是否正确相对于给定的断言而言。断言的描述用谓词或谓词公式。

部分正确

Definition 1.1 设 φ 和 ψ 是两个谓词。其中 φ 称为前断言， ψ 称为后断言。 T 在 I 的解释之下对于 φ 和 ψ 是部分正确的，当且仅当

$$\forall \sigma \in \Sigma, \varphi(\sigma) = true \wedge \mathcal{M}_I(T)(\sigma) \downarrow \rightarrow \psi(\mathcal{M}_I(T)(\sigma))。$$

例子

Example 1.1 证明程序 T_0 在解释 $I = (NAT, I_0)$ 下对于前断言 $x \geq 0$ 和后断言 $y_1 = \sqrt{x}$ 是部分正确的。

假设程序运行过程如下：

$(beg, \sigma_0)(test, \sigma_1)(loop, \sigma_2)(inloop, \sigma_3)(test, \sigma_4)(loop, \sigma_5) \cdots (test, \sigma_{n-1})(end, \sigma_n) \cdots$

这时迁移个数为 n ，即 $l_n = end$ 。我们需要证明的是 $(y_1 = \sqrt{x})(\sigma_n)$ 即 $\sigma_n(y_1) = \sqrt{\sigma_n(x)}$ 。

由于 $\sigma_n = \sigma_{n-1}$ 且 $(test, \sigma_{n-1}) \Rightarrow (end, \sigma_n)$ 则

$$\neg(\sigma_n(y_3) \leq \sigma_n(x))$$

因此，如果 $\varphi'(\sigma_{n-1})$ 成立且

$$\neg(\sigma_n(y_3) \leq \sigma_n(x)) \wedge \varphi'(\sigma_n) \rightarrow \sigma_n(y_1) = \sqrt{\sigma_n(x)}$$

则

$$\sigma_n(y_1) = \sqrt{\sigma_n(x)}$$

设 φ' 为

$$x = \sigma_0(x) \wedge y_1^2 \leq x \wedge y_2 = 2 * y_1 + 1 \wedge y_3 = (y_1 + 1)^2$$

$\neg(\sigma_n(y_3) \leq \sigma_n(x)) \wedge \varphi'(\sigma_n) \rightarrow \sigma_n(y_1) = \sqrt{\sigma_n(x)}$ 是成立的。我们需证 $\varphi'(\sigma_{n-1})$ ，即

$$\begin{aligned}\sigma_{n-1}(x) &= \sigma_0(x) \wedge \\ \sigma_{n-1}(y_1)^2 &\leq \sigma_{n-1}(x) \wedge \\ \sigma_{n-1}(y_2) &= 2 * \sigma_{n-1}(y_1) + 1 \wedge \\ \sigma_{n-1}(y_3) &= (\sigma_{n-1}(y_1) + 1)^2\end{aligned}$$

我们用归纳法证明以上公式。

由于 $n-1$ 时所对应的标号是 $test$ ，

我们只要证明对任意 k 当 σ_k 所对应的状态其标号为 $test$ 时

$$\begin{aligned}\sigma_k(x) &= \sigma_0(x) \wedge \\ \sigma_k(y_1)^2 &\leq \sigma_k(x) \wedge \\ \sigma_k(y_2) &= 2 * \sigma_k(y_1) + 1 \wedge \\ \sigma_k(y_3) &= (\sigma_k(y_1) + 1)^2\end{aligned}$$

成立。

- $k = 0$ 时，由于标号为 beg ，无须证明。
- $k = 1$ 时，标号为 $test$ 。由于 $\sigma_1(y_1) = 0, \sigma_1(y_2) = 1, \sigma_1(y_3) = 1, \sigma_1(x) = \sigma_0(x)$ 且 $\sigma_0(x) \geq 0$ 。

因此

$$\begin{aligned}\sigma_k(x) &= \sigma_0(x) \wedge \\ \sigma_k(y_1)^2 &\leq \sigma_k(x) \wedge \\ \sigma_k(y_2) &= 2 * \sigma_k(y_1) + 1 \wedge \\ \sigma_k(y_3) &= (\sigma_k(y_1) + 1)^2\end{aligned}$$

成立。

- 假设 $k \leq i$ 时, 目标成立。当 $k = i + 1$ 且 $i \geq 1$ 时, 若标号不为 *test* 则无须证明。若标号为 *test* 则 $k \geq 4$ 且

$$\begin{aligned} (test, \sigma_{k-3}) &\Rightarrow (loop, \sigma_{k-2}) \\ (loop, \sigma_{k-2}) &\Rightarrow (inloop, \sigma_{k-1}) \\ (inloop, \sigma_{k-1}) &\Rightarrow (test, \sigma_k) \end{aligned}$$

因此

$$\begin{aligned} \sigma_k &= \sigma_{k-1}[y_3/I(y_2 + y_3)](\sigma_{k-1}) \\ \sigma_{k-1} &= \sigma_{k-2}[y_1/I(y_1 + 1)](\sigma_{k-2})[y_2/I(y_2 + 2)](\sigma_{k-2}) \\ \sigma_{k-2} &= \sigma_{k-3} \wedge I(y_3 \leq x)(\sigma_{k-3}) \end{aligned}$$

因此

$$\begin{aligned} \sigma_k(x) &= \sigma_{k-3}(x) = \sigma_0(x) \\ (\sigma_k(y_1))^2 &= (\sigma_{k-3}(y_1) + 1)^2 = \sigma_{k-3}(y_3) \leq \sigma_{k-3}(x) = \sigma_k(x) \\ \sigma_k(y_2) &= \sigma_{k-3}(y_2) + 2 = 2 * \sigma_{k-2}(y_1) + 3 = 2 * \sigma_k(y_1) + 1 \\ \sigma_k(y_3) &= \sigma_{k-3}(y_2) + \sigma_{k-3}(y_3) + 2 = (\sigma_{k-3}(y_1) + 2)^2 = (\sigma_k(y_1) + 1)^2 \end{aligned}$$

因此对任意 k 当 σ_k 所对应的状态其标号为 *test* 时

$$\begin{aligned} \sigma_k(x) &= \sigma_0(x) \wedge \\ \sigma_k(y_1)^2 &\leq \sigma_k(x) \wedge \\ \sigma_k(y_2) &= 2 * \sigma_k(y_1) + 1 \wedge \\ \sigma_k(y_3) &= (\sigma_k(y_1) + 1)^2 \end{aligned}$$

成立。

终止

Definition 1.2 设 φ 和 ψ 是两个谓词。其中 φ 称为前断言, ψ 称为后断言。 T 在 I 的解释之下对于 φ 是终止的, 当且仅当

$$\forall \sigma \in \Sigma, \quad \varphi(\sigma) = true \rightarrow \mathcal{M}_I(T)(\sigma) \downarrow.$$

例子

Example 1.2 证明程序 T_0 在解释 $I = (NAT, I_0)$ 下对于前断言 *true* 是终止的。

假设程序不终止, 则程序的运行过程为

$$(l_0, \sigma_0)(l_1, \sigma_1)(l_2, \sigma_2)(l_3, \sigma_3)(l_4, \sigma_4)(l_5, \sigma_5) \cdots$$

其中 $l_0 = beg$, 对所有 $k \geq 0$ 有

$$l_{3k+1} = test, l_{3k+2} = loop, l_{3k+3} = inloop \text{ 且 } \sigma_{3k+1}(y_3) \leq x$$

以下我们证明对所有 $k \geq 0$

有 $\sigma_{3k+1}(y_3) \geq k$ 且 $\sigma_{3k+1}(x) = \sigma_0(x)$ 。

- 由于 $\sigma_1(y_3) = 1$ 且 $\sigma_1(x) = \sigma_0(x)$ 。
- 因此 $\sigma_{3*0+1}(y_3) \geq 0$ 且 $\sigma_{3*0+1}(x) = \sigma_0(x)$ 。

- 假设 $k = i$ 时有 $\sigma_{3i+1}(y_3) \geq i$ 且 $\sigma_{3i+1}(x) = \sigma_0(x)$,
我们需证 $k = i + 1$ 时有 $\sigma_{3(i+1)+1}(y_3) \geq i + 1$ 且
 $\sigma_{3(i+1)+1}(x) = \sigma_0(x)$.

根据前面的分析我们有

$$\begin{aligned}\sigma_{3(i+1)+1}(x) &= \sigma_{3(i+1)+1-3}(x) = \sigma_0(x) \\ \sigma_{3(i+1)+1}(y_3) &= \sigma_{3(i+1)+1-3}(y_2) + \sigma_{3(i+1)+1-3}(y_3) + 2 \geq i + 1\end{aligned}$$

因此 $k = i + 1$ 时有

$$\sigma_{3(i+1)+1}(y_3) \geq i + 1 \text{ 且 } \sigma_{3(i+1)+1}(x) = \sigma_0(x) .$$

因此对所有 $k \geq 0$ 有 $\sigma_{3k+1}(y_3) \geq k$ 且 $\sigma_{3k+1}(x) = \sigma_0(x)$.

取 $k = \sigma_0(x) + 1$, 则 $\sigma_{3k+1}(y_3) \leq \sigma_{3k+1}(x)$ 不成立, 与程序不终止的前提矛盾.

完全正确

Definition 1.3 设 φ 和 ψ 是两个谓词。其中 φ 称为前断言, ψ 称为后断言。 T 在 I 的解释之下对于 φ 和 ψ 是完全正确的, 当且仅当

$$\forall \sigma \in \Sigma, \varphi(\sigma) = true \rightarrow \mathcal{M}_I(T)(\sigma) \downarrow \wedge \psi(\mathcal{M}_I(T)(\sigma)) .$$

例子

Example 1.3 证明程序 T_0 在解释 $I = (NAT, I_0)$ 下对于前断言 $x \geq 0$ 和后断言 $y_1 = \sqrt{x}$ 是完全正确的。

首先证明引理: 对所有状态 $\sigma \in \Sigma$ 和所有 $0 \leq k \leq \sqrt{\sigma_0(x)}$,

$$(l_0 = beg, \sigma_0) \Rightarrow (l_{3k+1}, \sigma_{3k+1})$$

且

$$\begin{aligned}l_{3k+1} &= test \\ \sigma_{3k+1}(x) &= \sigma_0(x) \\ \sigma_{3k+1}(y_1) &= k \\ \sigma_{3k+1}(y_2) &= 2k + 1 \\ \sigma_{3k+1}(y_3) &= (k + 1)^2\end{aligned}$$

引理的证明使用归纳法如下:

- $k = 0$ 时, 显然引理成立。
- 假设 $k = i$ 且 $k \leq \sqrt{\sigma_0(x)}$ 时有

$$\begin{aligned}l_{3k+1} &= test \\ \sigma_{3k+1}(x) &= \sigma_0(x) \\ \sigma_{3k+1}(y_1) &= k \\ \sigma_{3k+1}(y_2) &= 2k + 1 \\ \sigma_{3k+1}(y_3) &= (k + 1)^2\end{aligned}$$

则 $k = i + 1$ 且 $k \leq \sqrt{\sigma_0(x)}$ 时有

$$\begin{aligned}l_{3(i+1)+1} &= test \\ \sigma_{3(i+1)+1}(x) &= \sigma_{3i+1}(x) = \sigma_0(x) \\ \sigma_{3(i+1)+1}(y_1) &= \sigma_{3i+1}(y_1) + 1 = i + 1 = k \\ \sigma_{3(i+1)+1}(y_2) &= \sigma_{3i+1}(y_2) + 2 = 2(i + 1) + 1 = 2k + 1 \\ \sigma_{3(i+1)+1}(y_3) &= \sigma_{3i+1}(y_3) + \sigma_{3i+1}(y_2) + 2 = (i + 2)^2 = (k + 1)^2\end{aligned}$$

因此引理成立。

设 $k = \sqrt{\sigma_0(x)}$

则 $\sigma_{3k+1}(y_3) = (k+1)^2 = (\sqrt{\sigma_0(x)} + 1)^2 > \sigma_0(x) = \sigma_{3k+1}(x)$ 。因此

$$(l_0 = beg, \sigma_0) \xrightarrow{*} (l_{3k+1}, \sigma_{3k+1}) \Rightarrow (end, \sigma_{3k+2})$$

且 $\sigma_{3k+2}(y_1) = \sigma_{3k+1}(y_1) = k = \sqrt{\sigma_0(x)}$ 。

相对于公式的正确性

Definition 1.4 设 φ 和 ψ 是两个公式。

- T 在 I 的解释之下对于 φ 和 ψ 是部分正确的, 记作

$$\models_I \{\varphi\}T\{\psi\}$$

当且仅当 T 在 I 的解释之下对于 $I(\varphi)$ 和 $I(\psi)$ 是部分正确的。

- T 在 I 的解释之下对于 φ 是终止的, 记作

$$\models_I [\varphi]T[true]$$

当且仅当 T 在 I 的解释之下对于 $I(\varphi)$ 是终止的。

- T 在 I 的解释之下对于 φ 和 ψ 是完全正确的, 记作

$$\models_I [\varphi]T[\psi]$$

当且仅当 T 在 I 的解释之下对于 $I(\varphi)$ 和 $I(\psi)$ 是完全正确的。

最弱前断言和最弱后断言

Definition 1.5 设 $T \in \mathcal{L}_1^B$ 为程序, φ 和 ψ 是两个谓词。其中 φ 称为前断言, ψ 称为后断言。给定解释 I 。

- φ 是 T 和 ψ 的最弱宽松前断言, 若

$$\varphi(\sigma) = true$$

当且仅当 $M_I(T)(\sigma) \uparrow \vee (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma)))$ 。

- φ 是 T 和 ψ 的最弱前断言, 若

$$\varphi(\sigma) = true$$

当且仅当 $M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma))$ 。

- 在 I 的解释之下, ψ 是 T 和 φ 的最弱后断言, 若

$$\psi(\sigma) = true$$

当且仅当存在状态 σ' 使得

$$\varphi(\sigma') = true \wedge M_I(T)(\sigma') = \sigma$$

1. (1a) T 对于 φ 和 ψ 是部分正确的。
 (1b) T 对于 φ' 和 ψ 是部分正确的, 则 $\varphi' \rightarrow \varphi$ 。
2. (2a) T 对于 φ 和 ψ 是完全正确的。
 (2b) T 对于 φ' 和 ψ 是完全正确的, 则 $\varphi' \rightarrow \varphi$ 。
3. (3a) T 对于 φ 和 ψ 是部分正确的。
 (3b) T 对于 φ 和 ψ' 是部分正确的, 则 $\psi \rightarrow \psi'$ 。

证明:

(1)

条件: $\varphi(\sigma) \leftrightarrow (M_I(T)(\sigma) \uparrow \vee (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma))))$

(1a) T 对于 φ 和 ψ 是部分正确的:

由条件 (1) 有

$\varphi(\sigma) \rightarrow (M_I(T)(\sigma) \uparrow \vee (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma))))$

因此

$\varphi(\sigma) \wedge M_I(T)(\sigma) \downarrow \rightarrow \psi(M_I(T)(\sigma))$

(1b)

T 对于 φ' 和 ψ 是部分正确的则 $\varphi' \rightarrow \varphi$:

前提即 $\varphi'(\sigma) \wedge (M_I(T)(\sigma) \downarrow \rightarrow \psi(M_I(T)(\sigma)))$

即 $\varphi'(\sigma) \rightarrow (M_I(T)(\sigma) \uparrow \vee (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma))))$

由条件 (1) 即 $\varphi'(\sigma) \rightarrow \varphi(\sigma)$

(2)

条件: $\varphi(\sigma) \leftrightarrow (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma)))$

(2a) T 对于 φ 和 ψ 是完全正确的:

由条件 (2) 就有

$\varphi(\sigma) \rightarrow (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma)))$

(2b) T 对于 φ' 和 ψ 是完全正确的, 则 $\varphi' \rightarrow \varphi$:

前提即 $\varphi'(\sigma) \rightarrow (M_I(T)(\sigma) \downarrow \wedge \psi(M_I(T)(\sigma)))$

由条件 (2) 即 $\varphi'(\sigma) \rightarrow \varphi(\sigma)$

(3)

条件: $\psi(\sigma') \leftrightarrow \exists \sigma''. (\varphi(\sigma'') \wedge M_I(T)(\sigma'') = \sigma')$

(3a) T 对于 φ 和 ψ 是部分正确的:

要证 $\varphi(\sigma) \wedge M_I(T)(\sigma) \downarrow \rightarrow \psi(M_I(T)(\sigma))$

即 $\varphi(\sigma) \wedge M_I(T)(\sigma) = \sigma' \rightarrow \psi(\sigma')$

由条件 (3) 即

$\varphi(\sigma) \wedge M_I(T)(\sigma) = \sigma' \rightarrow \exists \sigma''. (\varphi(\sigma'') \wedge M_I(T)(\sigma'') = \sigma')$

显然成立。

(3b) T 对于 φ 和 ψ' 是部分正确的, 则 $\psi \rightarrow \psi'$:

前提即 $\varphi(\sigma) \wedge (M_I(T)(\sigma) \downarrow \rightarrow \psi'(M_I(T)(\sigma)))$

即 $\varphi(\sigma) \wedge M_I(T)(\sigma) = \sigma' \rightarrow \psi'(\sigma')$

又由条件 (3) 有 $\psi(\sigma') \rightarrow \exists \sigma''. (\varphi(\sigma'') \wedge M_I(T)(\sigma'') = \sigma')$

因此 $\psi(\sigma') \rightarrow \psi'(\sigma')$

Theorem 1.1 设 $T \in \mathcal{L}_1^B$ 为程序, φ 和 ψ 是两个谓词。

- 设 φ 是 T 和 ψ 的最弱宽松前断言。则

对所有 φ' , T 对于 φ' 和 ψ 是部分正确的当且仅当 $\varphi' \rightarrow \varphi$ 。

- 设 φ 是 T 和 ψ 的最弱前断言。则
对所有 φ' , T 对于 φ' 和 ψ 是完全正确的当且仅当 $\varphi' \rightarrow \varphi$ 。
- 设 ψ 是 T 和 φ 的最强后断言。则
对所有 ψ' , T 对于 φ 和 ψ' 是部分正确的当且仅当 $\psi \rightarrow \psi'$ 。

习题

设

$$B = (\{x, y, n, a\}, \{0, 1, 2, 3, \dots, +, -, *\}, \{<, =, >\})$$

给定以下 \mathcal{L}_1^B 中的程序 T :

```

beg:  (x, y) := (0, 0) goto l1
l1:  if (x < n) goto l2 else goto l3
l2:  (y, x) := (y + x * (x + 1), x + 1) goto l1
l3:  (y) := (3 * y) goto end

```

给定 I 为 B 在整数上的正常解释。证明以下命题成

立:

-
- (1) T 对于前断言 $n \geq 0$ 和后断言 $y = n * n * n - n$ 部分正确。
 - (2) T 对于前断言 $n \geq 0$ 能够终止。
 - (3) T 对于前断言 $n \geq 0$ 和后断言 $y = n * n * n - n$ 完全正确。
-

§1.2.2 基于路径的推理

上一节的证明基于对程序过程的分析。首先确定了程序的运行过程为如下序列:

$$(beg = l_0, \sigma_0)(test, \sigma_1)(loop, \sigma_2)(inloop, \sigma_3)(test, \sigma_4)(loop, \sigma_5) \cdots (test, \sigma_{n-1})(end, \sigma_n) \cdots$$

这个例子很简单, 只有一个循环。一般来讲, 确定程序运行过程有一定的难度。

Example 1.4 带有两个循环的程序。记为 T_2 。

```

beg:      (y1, y2) := (x, 1); goto test-1;
test-1:   if y1 > 100 then goto test-2 else upd-1 fi;
test-2:   if y2 ≠ 1 then goto upd-2 else res fi;
upd-1:    (y1, y2) := (y1 + 11, y2 + 1); goto test-1;
upd-2:    (y1, y2) := (y1 - 10, y2 - 1); goto test-1;
res:      z := y1 - 10; goto end
    
```

以上程序有两个循环。由于它们相互作用, 不可能用一个或几个带循环次数参数的序列来表示。虽然简单地罗列程序运行过程有一定的难度, 但如果我们略去运行过程中的具体变量状态, 我们可以将运行过程表示如下

$$beg, (test_1, ((upd_1)|(test_2, upd_2))^*, test_1, test_2, res, end$$

但如果我们有方法分析一些关键的标号之间的关系和证明某些性质在一些关键的标号时成立, 我们就可以证明程序所具有的性质。以上运行过程也可以写成

$$(beg), ((test_1, upd_1)^*, (test_1, test_2, upd_2)^*)^*, (test_1, test_2, res, end)$$

我们可以考虑以下标号序列

```

(beg, test1)
(test1, test2, upd2, test1)
(test1, upd1, test1)
(test1, test2, res, end)
    
```

我们将标号序列称为路径。假设

$$\begin{aligned}
 &\forall \sigma. (\varphi(\sigma) \wedge (beg, \sigma) \Rightarrow (test_1, \sigma') \rightarrow \zeta(\sigma')) \\
 &\forall \sigma. (\zeta(\sigma) \wedge (test_1, \sigma) \Rightarrow (test_2, \sigma') \Rightarrow (upd_2, \sigma'') \Rightarrow (test_1, \sigma''') \rightarrow \zeta(\sigma''')) \\
 &\forall \sigma. (\zeta(\sigma) \wedge (test_1, \sigma) \Rightarrow (upd_1, \sigma') \Rightarrow (test_1, \sigma'') \rightarrow \zeta(\sigma'')) \\
 &\forall \sigma. (\zeta(\sigma) \wedge (test_1, \sigma) \Rightarrow (test_2, \sigma') \Rightarrow (res, \sigma'') \Rightarrow (end, \sigma''') \rightarrow \psi(\sigma'''))
 \end{aligned}$$

则 $\{\varphi\}T_2\{\psi\}$ 成立。

对于这样的分析, 我们有两个方面需要继续讨论的。

第一, 挑选路径的方法, 即怎样保证我们考虑了所有应该考虑的路径。

第二, 路径正确性的证明方法, 即怎样证明路径对于给定的公式是正确的。

我们考虑以下方法。首先我们给出几个定义。

Definition 1.6 设 $\alpha = (l_0, l_1, \dots, l_n)$ 为一路径。

$$\mathcal{M}_I(\alpha)(\sigma_0) = \begin{cases} \sigma_n & \text{若存在 } \{\sigma_1, \dots, \sigma_n\} \subseteq \Sigma \\ & \text{使得 } (l_0, \sigma_0) \Rightarrow (l_1, \sigma_1) \Rightarrow \dots \Rightarrow (l_n, \sigma_n) 。 \\ \text{无定义} & \text{若不存在这样的状态集。} \end{cases}$$

若 $l_0 = \text{begin}$ 且 $l_n = \text{end}$ 则 $\alpha = (l_0, l_1, \dots, l_n)$ 称为完整路径。 $\mathcal{M}_I(T)(\sigma) = \sigma'$ 当且仅当存在完整路径 α 使得 $\mathcal{M}_I(T)(\alpha) = \sigma'$ 。

Definition 1.7 设 p 和 q 是两个公式。

- α 在 I 的解释之下对于 p 和 q 是部分正确的, 记作 $\models_I \{p\}\alpha\{q\}$, 若 $\forall \sigma \in \Sigma, I(p)(\sigma) = \text{true} \wedge \mathcal{M}_I(\alpha)(\sigma) \downarrow \rightarrow I(q)(\mathcal{M}_I(\alpha)(\sigma))$ 。
- α 在 I 的解释之下对于 p 是终止的, 记作 $\models_I [p]\alpha[\text{true}]$, 若 $\forall \sigma \in \Sigma, I(p)(\sigma) = \text{true} \rightarrow \mathcal{M}_I(\alpha)(\sigma) \downarrow$ 。
- α 在 I 的解释之下对于 p 和 q 是完全正确的, 记作 $\models_I [p]\alpha[q]$, 若 $\forall \sigma \in \Sigma, I(p)(\sigma) = \text{true} \rightarrow \mathcal{M}_I(\alpha)(\sigma) \downarrow \wedge I(q)(\mathcal{M}_I(\alpha)(\sigma))$ 。

Definition 1.8 设 α 为路径, p 和 q 是两个公式。

- p 是 α 和 q 的最弱宽松前断言, 若 $I(p)(\sigma) = \text{true}$ 当且仅当 $\mathcal{M}_I(\alpha)(\sigma)$ 无定义或 $\mathcal{M}_I(\alpha)(\sigma)$ 有定义且 $I(q)(\mathcal{M}_I(\alpha)(\sigma))$ 成立。
- p 是 α 和 q 的最弱前断言, 若 $I(p)(\sigma) = \text{true}$ 当且仅当 $\mathcal{M}_I(\alpha)(\sigma)$ 有定义且 $I(q)(\mathcal{M}_I(\alpha)(\sigma))$ 成立。
- q 是 α 和 p 的最强后断言, 若 $I(q)(\sigma) = \text{true}$ 当且仅当存在状态 σ' 使得 $I(p)(\sigma') = \text{true}$ 且 $\mathcal{M}_I(\alpha)(\sigma') = \sigma$ 。

Definition 1.9 设 $\alpha = (l_0, \dots, l_k)$ 为一路径。 $wlp(\alpha, q)$ 定义如下。

- $k = 0$ 则 $wlp(\alpha, q) = q$ 。
- $k = 1$ 时, 若 $l_0: (v_1, v_2, \dots, v_n) := (e_1, e_2, \dots, e_n) \text{ goto } l_1$ 为 T 中的指令, 则

$$wlp(\alpha, q) = q[v_1/e_1] \cdots [v_n/e_n].$$

若 $l_0: \text{if } (b) \text{ goto } l \text{ else goto } l'$, 为 T 中的指令, 则

$$wlp(\alpha, q) = \begin{cases} b \rightarrow q & \text{若 } l = l_1; \\ \neg b \rightarrow q & \text{若 } l' = l_1. \end{cases}$$

- $k > 1$ 时, 设 $\alpha_0 = (l_0, l_1)$, $\alpha_1 = (l_1, \dots, l_k)$ 。则 $wlp(\alpha, q) = wlp(\alpha_0, wlp(\alpha_1, q))$ 。

Theorem 1.2 $I(wlp(\alpha, q))$ 是路径 α 和 q 的最弱宽松前断言。

要证明

$$I(wlp(\alpha, q))(\sigma) = true \leftrightarrow M_I(\alpha)(\sigma) \uparrow \vee (M_I(\alpha)(\sigma) \downarrow \wedge I(q)(M_I(\alpha)(\sigma)))$$

设 $\alpha = (l_0, l_1, \dots, l_n)$ 为一路径。

(1) $n = 0$.

(2) $n = 1$.

(2a) $l_0: (v_1, v_2, \dots, v_n) := (e_1, e_2, \dots, e_n) \text{ goto } l_1$

$$wlp(\alpha, q) = q[v_1/e_1] \cdots [v_n/e_n].$$

$$M_I(\alpha)(\sigma) = \sigma[v_1/e_1] \cdots [v_n/e_n]$$

(2b) $l_0: \text{if } (b) \text{ goto } l \text{ else goto } l'$,

(2b1) $l = l_1$

$$wlp(\alpha, q) = b \rightarrow q$$

(a) $I(b)(\sigma) = true$

$$M_I(\alpha)(\sigma) = \sigma$$

(b) $I(b)(\sigma) = false$

(2b2) $l' = l_1$

$$wlp(\alpha, q) = \neg b \rightarrow q$$

(a) $I(b)(\sigma) = true$

(b) $I(b)(\sigma) = false$

$$M_I(\alpha)(\sigma) = \sigma$$

(3) $\alpha_0 = (l_0, l_1)$, $\alpha_1 = (l_1, \dots, l_k)$ 。

$$wlp(\alpha, q) = wlp(\alpha_0, wlp(\alpha_1, q))$$

要证

$$I(wlp(\alpha_0, wlp(\alpha_1, q))) (\sigma) = true \leftrightarrow M_I(\alpha)(\sigma) \uparrow \vee (M_I(\alpha)(\sigma) \downarrow \wedge I(q)(M_I(\alpha)(\sigma)))$$

有

$$I(wlp(\alpha_0, wlp(\alpha_1, q))) (\sigma) = true \leftrightarrow M_I(\alpha_0)(\sigma) \uparrow \vee (M_I(\alpha_0)(\sigma) \downarrow \wedge I(wlp(\alpha_1, q))(M_I(\alpha_0)(\sigma)))$$

(1) $M_I(\alpha_0)(\sigma) \uparrow$ 则两个右边等价。

(2) $M_I(\alpha_0)(\sigma) \downarrow$ 要证

$$M_I(\alpha)(\sigma) \uparrow \vee (M_I(\alpha)(\sigma) \downarrow \wedge I(q)(M_I(\alpha)(\sigma))) \leftrightarrow I(wlp(\alpha_1, q))(M_I(\alpha_0)(\sigma))$$

即

$$M_I(\alpha)(\sigma) \uparrow \vee (M_I(\alpha)(\sigma) \downarrow \wedge I(q)(M_I(\alpha)(\sigma)))$$

\leftrightarrow

$$(M_I(\alpha_1)(M_I(\alpha_0)(\sigma)) \uparrow \vee (M_I(\alpha_1)(M_I(\alpha_0)(\sigma)) \downarrow \wedge I(q)(M_I(\alpha_1)(M_I(\alpha_0)(\sigma))))$$

显然成立

Definition 1.10 设 $\alpha = (l_0, \dots, l_k)$ 为一路径。

$$vc(p, \alpha, q) = p \rightarrow wlp(\alpha, q)$$

Theorem 1.3 若 $\models_I vc(p, \alpha, q)$ 则 $\models_I \{p\}\alpha\{q\}$ 。

要证明

$$I(vc(p, \alpha, q))(\sigma) = true \rightarrow I(p) \wedge M_I(\alpha)(\sigma) \downarrow \rightarrow I(q)(M_I(\alpha)(\sigma))$$

Definition 1.11 设 T 为程序, C 为标号集合。定义 $\gamma(T, C)$ 为满足以下条件的路径集合: $(l_0, \dots, l_k) \in \gamma(T, C)$ 当且仅当 (l_0, \dots, l_k) 是 T 的路径, $k > 0$, $l_0, l_k \in C$ 且 $l_1, \dots, l_{k-1} \notin C$ 。

Theorem 1.4 设 C 是标号集合, $beg, end \in C$, T 的每个循环至少有一个标号包含于 C 且 C 中的每个标号 l 有一个对应的公式 q_l 。若

$$\forall \alpha = (l_0, \dots, l_k) \in \gamma(T, C), \models_I vc(q_{l_0}, \alpha, q_{l_k})$$

则

$$\models \{q_{beg}\}T\{q_{end}\}$$

证明

我们先证明一个更为一般的结论, 即

若 $l_a, l_b \in C$ 且 $\alpha = (l_a, \dots, l_b)$ 为 T 的路径,

则 $\models \{q_{l_a}\}\alpha\{q_{l_b}\}$ 。

我们假设定理的前提成立。

设 $len(l_a, \dots, l_b)$ 为 (l_a, \dots, l_b) 中 C 中元素的个数。

我们用归纳法证明 $\models \{q_{l_a}\}\alpha\{q_{l_b}\}$ 。

- 若 $len(l_a, \dots, l_b) = 2$ 则根据假设 $\models_I vc(q_{l_a}, (l_a, \dots, l_b), q_{l_b})$ 成立,

因此 $\models_I \{q_{l_a}\}(l_a, \dots, l_b)\{q_{l_b}\}$ 成立。

- 假设对任意 $l_a, l_b \in C$, $len(l_a, \dots, l_b) \leq k$ 时
 $\models_I \{q_{l_a}\}(l_a, \dots, l_b)\{q_{l_b}\}$ 成立。
 对任意 l_a, l_b , $len(l_a, \dots, l_b) \leq k + 1$ 时,
 我们可将 (l_a, \dots, l_b) 分成两段 (l_a, \dots, l_c) 和 (l_c, \dots, l_b) ,
 其中 $l_c \in C$ 且 $len(l_a, \dots, l_c), len(l_c, \dots, l_b) \leq k$ 。
 根据假设我们有 $\models_I \{q_{l_a}\}(l_a, \dots, l_c)\{q_{l_c}\}$ 和 $\models_I \{q_{l_c}\}(l_c, \dots, l_b)\{q_{l_b}\}$ 。

因此 $\models_I \{q_{l_a}\}(l_a, \dots, l_b)\{q_{l_b}\}$ 成立。

根据归纳法对任意 $l_a, l_b \in C$, $\models_I \{q_{l_a}\}(l_a, \dots, l_b)\{q_{l_b}\}$ 成立。由于 $l_0 = beg \in C, l_n = end \in C$,
 因此 $\models_I \{q_{beg}\}(l_0, \dots, l_n)\{q_{end}\}$ 成立。
 因此对所有以 beg, end 为开始和终结点的路径 α , $\models_I \{q_{beg}\}\alpha\{q_{end}\}$ 成立。

例子

Example 1.5 设 $I = (NAT, I_0)$ 。用归纳断言方法证明
 $\{x = c\}T_1\{y_1 = \sqrt{c}\}$ 。

证明分三个步骤。

- 确定标号集合 $C = \{beg, test, end\}$ 。
- 挑选断言 $q_{beg}, q_{end}, q_{test}$

$$\begin{aligned} q_{beg} \quad & x = c \\ q_{end} \quad & y_1 = \sqrt{c} \\ q_{test} \quad & x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \end{aligned}$$

- 确定需要证明的路径

$$\begin{aligned} & (beg, test) \\ & (test, loop, inloop, test) \\ & (test, end) \end{aligned}$$

- 证明路径正确性

$$\begin{aligned} & \models_I vc(q_{beg}, (beg, test), q_{test}) \\ & \models_I vc(q_{test}, (test, loop, inloop, test), q_{test}) \\ & \models_I vc(q_{test}, (test, end), q_{end}) \end{aligned}$$

Example 1.6 设 $I = (NAT, I_0)$ 。用归纳断言方法证明
 $\{x \leq 100\}T_2\{z = 91\}$ 。

证明分三个步骤。

- 确定标号集合 $C = \{beg, test_1, end\}$ 。
- 挑选断言 $q_{beg}, q_{end}, q_{test_1}$

$$\begin{aligned} q_{beg} \quad & x \leq 100 \\ q_{end} \quad & z = 91 \\ q_{test} \quad & y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101) \end{aligned}$$

- 确定需要证明的路径

$(beg, test_1)$

$(test_1, test_2, udp_2, test_1)$

$(test_1, udp_1, test_1)$

$(test_1, test_2, res, end)$

- 证明路径正确性

$\models_I vc(q_{beg}, (beg, test_1), q_{test_1})$

$\models_I vc(q_{test_1}, (test_1, test_2, udp_2, test_1), q_{test_1})$

$\models_I vc(q_{test_1}, (test_1, udp_1, test_1), q_{test_1})$

$\models_I vc(q_{test_1}, (test_1, test_2, res, end), q_{end})$

Theorem 1.5 设

- C 是标号集合, $beg \in C$, T 的每个循环至少有一个标号包含于 C 且 C 中的每个标号 l 有一个对应的公式 q_l
- (W, \sqsubseteq) 是一 WFS
- $C' \subseteq C$ 是标号集合, T 的每个循环至少有一个标号包含于 C' 且 C' 中的每个标号 l 有一个对应的函数 $g_l : \Sigma \rightarrow W$

若以下条件成立则 $\models [q_{beg}]T[true]$:

$$\forall \alpha \in \gamma(T, C), \models_I vc(q_{l_0}, \alpha, q_{l_k})$$

$$\forall \alpha \in \gamma(T, C'), I(q_{l_0})(\sigma) = true \wedge M_I(\alpha)(\sigma) \downarrow \rightarrow g_{l_k}(M_I(\alpha)(\sigma)) \sqsubseteq g_{l_0}(\sigma)$$

例子

Example 1.7 证明程序 T_1 在解释 $I = (NAT, I_0)$ 下对于前断言 $x = c \wedge c \geq 0$ 是终止的。

证明。

- 确定标号集合 $C = \{beg, test\}$ 。
- 挑选断言 q_{beg}, q_{test}

$$q_{beg} \quad x = c \wedge c \geq 0$$

$$q_{test} \quad x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1$$

- 挑选 $(W, \sqsubseteq) = (\{0, 1, 2, \dots\}, \leq)$
- 确定标号集合 $C' = \{test\}$ 。
- 挑选项 $g_{test} : \Sigma \rightarrow W$

$$g_{test}(\sigma) = \sigma(x) + 1 - \sigma(y_3)$$

- 确定需要证明的路径

$$(beg, test)$$

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$\models_I vc(q_{beg}, (beg, test), q_{test})$$

$$\models_I vc(q_{test}, (test, loop, inloop, test), q_{test})$$

- 确定需要证明的路径

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$I(q_{test})(\sigma) = true \wedge M_I(test, loop, inloop, test)(\sigma) \downarrow$$

→

$$g_{test}(M_I(test, loop, inloop, test)(\sigma)) < g_{test}(\sigma)$$

即（简单起见省略了符号 σ ）

$$x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge (y_3 \leq x)$$

→

$$x + 1 - (y_2 + y_3 + 2) < x + 1 - y_3$$

Example 1.8 证明程序 T_1 在解释 $I = (INT, I_0)$ 下对于前断言 $x = c \wedge c \geq 0$ 是终止的。

证明。

- 确定标号集合 $C = \{beg, test\}$ 。
- 挑选断言 q_{beg}, q_{test}

$$q_{beg} \quad x = c \wedge c \geq 0$$

$$q_{test} \quad y_2 \geq 0$$

- 挑选 $(W, \sqsubseteq) = (\{0, 1, 2, \dots\}, \leq)$
- 确定标号集合 $C' = \{test\}$ 。
- 挑选项 $g_{test} : \Sigma \rightarrow W$

$$g_{test}(\sigma) = \begin{cases} \sigma(x) + 1 - \sigma(y_3) & \text{若 } \sigma(y_3) \leq \sigma(x) \\ 0 & \text{否则。} \end{cases}$$

- 确定需要证明的路径

$$(beg, test)$$

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$\models_I vc(q_{beg}, (beg, test), q_{test})$$

$$\models_I vc(q_{test}, (test, loop, inloop, test), q_{test})$$

- 确定需要证明的路径

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$I(q_{test})(\sigma) = true \wedge M_I(test, loop, inloop, test)(\sigma) \downarrow$$

→

$$g_{test}(M_I(test, loop, inloop, test)(\sigma)) < g_{test}(\sigma)$$

即

$$y_2 \geq 0 \wedge (y_3 \leq x)$$

$$\rightarrow x + 1 - (y_2 + y_3 + 2) < x + 1 - y_3$$

或

$$y_2 \geq 0 \wedge (y_3 \leq x)$$

$$\rightarrow 0 < x + 1 - y_3$$

Theorem 1.6 设

- C 是标号集合, $beg \in C$, T 的每个循环至少有一个标号包含于 C 且 C 中的每个标号 l 有一个对应的公式 q_l
- $(W, I_0(\sqsubseteq))$ 是一 WFS 且 $\sqsubseteq \in P$
- w 为最多有一个自由变量的公式且

$$W = \{\sigma(x) \mid I(w)(\sigma) = true\}$$

- $C' \subseteq C$ 是标号集合, T 的每个循环至少有一个标号包含于 C' , C' 中的每个标号 l 有一个对应的项 t_l 且 $\models_I q_l \rightarrow w[x/t_l]$ 。

若以下条件成立则 $\models [q_{beg}]T[true]$:

$$\forall \alpha \in \gamma(T, C), \models_I vc(q_{l_0}, q_{l_k})$$

$$\forall \alpha \in \gamma(T, C'), \models_I vc(q_{l_0} \wedge t_{l_0} = a, \alpha, t_{l_k} \sqsubseteq a)$$

例子

Example 1.9 证明程序 T_1 在解释 $I = (NAT, I_0)$ 下对于前断言 $x = c \wedge c \geq 0$ 是终止的。

证明。

- 确定标号集合 $C = \{beg, test\}$ 。
- 挑选断言 q_{beg}, q_{test}

$$q_{beg} \quad x = c \wedge c \geq 0$$

$$q_{test} \quad x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1$$

- 挑选 $(W, \sqsubseteq) = (\{0, 1, 2, \dots\}, \leq)$
- 挑选 $w = true$ 并证明 $W = \{\sigma(x) \mid I(w)(\sigma) = true\}$
- 确定标号集合 $C' = \{test\}$ 。
- 挑选项 $t_{test} = x + 1 - y_3$ 并证明 $q_{test} \rightarrow w_x^{t_{test}}$
- 确定需要证明的路径

$$(beg, test)$$

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$\models_I vc(q_{beg}, (beg, test), q_{test})$$

$$\models_I vc(q_{test}, (test, loop, inloop, test), q_{test})$$

- 确定需要证明的路径

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$\models_I vc(q_{test} \wedge t_{test} = v, (test, loop, inloop, test), t_{test} < v)$$

即

$$\begin{aligned} x &= c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge x + 1 - y_3 = v \\ &\rightarrow \\ &((y_3 \leq x) \rightarrow x + 1 - (y_2 + y_3 + 2) < v) \end{aligned}$$

Example 1.10 证明程序 T_1 在解释 $I = (INT, I_0)$ 下对于前断言 $x = c \wedge c \geq 0$ 是终止的。

证明。

- 确定标号集合 $C = \{beg, test\}$ 。
- 挑选断言 q_{beg}, q_{test}

$$\begin{aligned} q_{beg} \quad x &= c \wedge c \geq 0 \\ q_{test} \quad x &= c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge y_2 \geq 1 \end{aligned}$$

- 挑选 $(W, \sqsubseteq) = (\{0, 1, 2, \dots\}, \leq)$
- 挑选 $w = (x \geq 0)$ 并证明 $W = \{\sigma(x) | I(w)(\sigma) = true\}$
- 确定标号集合 $C' = \{test\}$ 。
- 挑选项 $t_{test} = x + 1 - y_3 + y_2$ 并证明 $q_{test} \rightarrow w_x^{t_{test}}$
- 确定需要证明的路径

$$\begin{aligned} &(beg, test) \\ &(test, loop, inloop, test) \end{aligned}$$

- 证明路径正确性

$$\begin{aligned} &\models_I vc(q_{beg}, (beg, test), q_{test}) \\ &\models_I vc(q_{test}, (test, loop, inloop, test), q_{test}) \end{aligned}$$

- 确定需要证明的路径

$$(test, loop, inloop, test)$$

- 证明路径正确性

$$\models_I vc(q_{test} \wedge t_{test} = v, (test, loop, inloop, test), t_{test} < v)$$

即

$$\begin{aligned} x &= c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge y_2 \geq 1 \wedge x + 1 - y_3 + y_2 = v \\ &\rightarrow \\ &((y_3 \leq x) \rightarrow x + 1 - (y_2 + y_3 + 2) + (y_2 + 2) < v) \end{aligned}$$

习题

设

$$B = (\{x, y, n, a\}, \{0, 1, 2, 3, \dots, +, -, *\}, \{<, =, >\})$$

给定以下 \mathcal{L}_1^B 中的程序 T :

```

beg:  (i, j, k, l) := (1, 0, 0, 1) goto l1
l1:   if  $\neg(x = y)$  goto l2 else goto end
l2:   if  $(x > y)$  goto l3 else goto l4
l3:   (x, i, j) := (x - y, i - k, j - l) goto l1
l4:   (y, k, l) := (y - x, k - i, l - j) goto l1

```

给定 I 为 B 在整数上的正常解释。证明以下命题成立:

-
- (1) T 对于前断言 $x = a \wedge y = b \wedge a \geq 0 \wedge b \geq 0$ 和后断言 $x = \gcd(a, b) \wedge x = i * a + j * b$ 部分正确。
- (2) T 对于前断言 $x = a \wedge y = b \wedge a \geq 0 \wedge b \geq 0$ 能够终止。
-

§1.3 结构化程序的推理

流程图程序的缺点:

循环的入口和出口不规范

从写程序的角度讲, 容易出错。

从证明的角度讲, 推理比较复杂。

从程序结构来讲, 可组合性较差。

结构化程序可以克服这些缺点。

这里定义的结构化程序是一阶逻辑的扩充。

给定函数符号集 F 和谓词符号集 P ,

我们可以在 $B = (F, P)$ 上建立一个一阶逻辑。

在此基础上, 我们增加以下集合的符号:

$$\{:=, ;, if, then, else, fi, while, do, od\}$$

结构化程序的语法定义如下。

$$A ::= x := t$$

$$S ::= A \mid S; S \mid \text{if } e \text{ then } S \text{ else } S \text{ fi} \mid \text{while } e \text{ do } S \text{ od}$$

基于 B 的结构化程序的集合记为 \mathcal{L}_2^B 。

例子

以下是一个计算阶乘的结构化程序, 记为 T_{jc} 。

```
y := 1; while  $x > 0$  do y := y * x; x := x - 1 od
```

给定 $a, b \in NAT$ 。定义 $\gcd(a, b)$ 如下。

若 $a, b > 0$ 则 $\gcd(a, b)$ 为 a, b 的最大公约数; 否则 $\gcd(a, b)$ 没定义。

以下是一个计算 $\gcd(x, y)$ 的程序, 记为 T_{gcd} 。

while $\neg(x = y)$ do if $(x > y)$ then $x := x - y$ else $y := y - x$ fi od

§1.3.1 操作语义

结构化程序 $T \in \mathcal{L}_2^B$ 的语义取决于 B 的解释。

给定 B 的一个解释 I 。

结构化程序的综合状态由程序的剩余部分和变量状态决定。

设 Σ 为变量状态的集合。

程序的综合状态集合为 $(\mathcal{L}_2^B \times \Sigma) \cup \Sigma$ 。

程序的迁移关系 \Rightarrow 为 $(\mathcal{L}_2^B \times \Sigma) \times ((\mathcal{L}_2^B \times \Sigma) \cup \Sigma)$ 的一个子集。

(1) $(S_0, \sigma_0) \Rightarrow (S_1, \sigma_1)$ 当且仅当以下一项成立。

S_0	条件	S_1	σ_1
$x := t; S$		S	$\sigma_0[x/I(t)(\sigma_0)]$
if (e) then S else S' fi	$\sigma_0 \models_I e$	S	σ_0
if (e) then S else S' fi	$\sigma_0 \not\models_I e$	S'	σ_0
if (e) then S else S' fi; S''	$\sigma_0 \models_I e$	$S; S''$	σ_0
if (e) then S else S' fi; S''	$\sigma_0 \not\models_I e$	$S'; S''$	σ_0
while (e) do S od	$\sigma_0 \models_I e$	$S; \text{while } (e) \text{ do } S \text{ od}$	σ_0
while (e) do S od	$\sigma_0 \not\models_I e$	-	-
while (e) do S od; S'	$\sigma_0 \models_I e$	$S; \text{while } (e) \text{ do } S \text{ od}; S'$	σ_0
while (e) do S od; S'	$\sigma_0 \not\models_I e$	S'	σ_0

(2) $(S_0, \sigma_0) \Rightarrow \sigma_1$ 当且仅当以下一项成立。

S_0	条件	σ_1
$x := t$		$\sigma_0[x/I(t)(\sigma_0)]$
while (e) do S od	$\sigma_0 \not\models_I e$	σ_0

程序 T 在初始变量状态为 σ 时的运行终止，当且仅当存在 σ' 使得 $(T, \sigma) \Rightarrow^* \sigma'$ 。程序的语义函数是从状态到状态的函数 $\mathcal{M}_I(T) : \Sigma \leftrightarrow \Sigma$ ，定义如下。

$$\mathcal{M}_I(T)(\sigma) = \begin{cases} \sigma' & \text{若 } (T, \sigma) \Rightarrow^* \sigma'。 \\ \text{无定义} & \text{若程序在初始变量状态为 } \sigma \text{ 时的运行不终止。} \end{cases}$$

程序的正确性描述可以是初始状态和终止状态的关系，以及程序是否终止。

比如，我们可以要求程序 T_{jc} 满足

$$\mathcal{M}_I(T)(\sigma)(y) = \sigma(x)!$$

或者要求程序满足

$$\sigma(x) > 0 \text{ 则 } \mathcal{M}_I(T)(\sigma)(y) = \sigma(x)!。$$

§1.3.2 指称语义

结构化程序的优点是它具有组合性。

比如给定程序 T_1 和 T_2 ，我们可将其组合成 $T_1; T_2$ ，再给一个公式 e ，我们可将其组合成 $\text{if } (e) \text{ then } T_1 \text{ else } T_2 \text{ fi}$ 等。

这样， $T_1; T_2$ 的语义就可以建立在 T_1 和 T_2 的语义上。

比如 $\mathcal{M}_I(T_1; T_2)(\sigma) = \mathcal{M}_I(T_2)\mathcal{M}_I(T_1)(\sigma)$ 。

这理有一个问题，即如果 $\mathcal{M}_I(T_1)(\sigma)$ 没定义，这么写就不合适。

为了方便这种组合，我们对 \mathcal{M} 进行扩充，记作 \mathcal{M}^ω ，其类型为 $\mathcal{M}_I^\omega(T) : \Sigma_\omega \rightarrow \Sigma_\omega$ ，

其中 $\Sigma_\omega = \Sigma \cup \{\omega\}$ 。 $\mathcal{M}_I^\omega(T)$ 可以定义如下。

$\mathcal{M}_I^\omega(x := t)(\sigma)$	$=$	$\begin{cases} \omega & \text{若 } \sigma = \omega \\ \mathcal{M}_I(x := t)(\sigma) = \sigma[x/I(t)(\sigma)] & \text{若 } \sigma \neq \omega \end{cases}$
$\mathcal{M}_I^\omega(T_1; T_2)(\sigma)$	$=$	$\begin{cases} \omega & \text{若 } \sigma = \omega \\ \mathcal{M}_I^\omega(T_2)\mathcal{M}_I^\omega(T_1)(\sigma) & \text{若 } \sigma \neq \omega \end{cases}$
$\mathcal{M}_I^\omega(\text{if } (e) \text{ then } T_1 \text{ else } T_2 \text{ fi})(\sigma)$	$=$	$\begin{cases} \omega & \text{若 } \sigma = \omega \\ \text{ite}(I(e)(\sigma), \mathcal{M}_I^\omega(T_1)(\sigma), \mathcal{M}_I^\omega(T_2)(\sigma)) & \text{若 } \sigma \neq \omega \end{cases}$
$\mathcal{M}_I^\omega(\text{while } (e) \text{ do } T_1 \text{ od})$	$=$	$\mu\Phi$

其中 $\Phi : [\Sigma_\omega \rightarrow \Sigma_\omega] \rightarrow [\Sigma_\omega \rightarrow \Sigma_\omega]$ 定义如下

$$\Phi(f)(\sigma) = \begin{cases} \omega & \text{若 } \sigma = \omega \\ \text{ite}(I(e)(\sigma), f(\mathcal{M}_I^\omega(T_1)(\sigma)), \sigma) & \text{若 } \sigma \neq \omega \end{cases}$$

首先，我们要证明这个定义是合适的，即等式右边公式的值是 Σ_ω 的元素。这可以由这些公式是连续函数来证明。

基于 $\mathcal{M}_I^\omega(T)$ ，我们可以定义 $\mathcal{M}_I(T) : \Sigma \leftrightarrow \Sigma$ 如下。

$$\mathcal{M}_I(T)(\sigma) = \begin{cases} \mathcal{M}_I^\omega(T)(\sigma) & \text{若 } \mathcal{M}_I^\omega(T)(\sigma) \neq \omega \\ \text{无定义} & \text{若 } \mathcal{M}_I^\omega(T)(\sigma) = \omega \end{cases}$$

这个定义和操作语义中对 $\mathcal{M}_I(T)$ 的定义是一致的。反过来，我们有

$$\mathcal{M}_I^\omega(T)(\sigma) = \begin{cases} \omega & \text{若 } \sigma = \omega \\ \omega & \text{若 } \mathcal{M}_I(T)(\sigma) \text{ 无定义} \\ \sigma' & \text{若 } \mathcal{M}_I(T)(\sigma) = \sigma' \end{cases}$$

Example 1.11 证明若 $\mathcal{M}_I(T_{jc})(\sigma)$ 有定义，则 $\mathcal{M}_I(T_{jc})(\sigma)(y) = \sigma(x)!$ 。

证明：只需证明 $\mathcal{M}_I^\omega(T_{jc})(\sigma)(y) \sqsubseteq (\sigma)(x)!$ 。

由于有

$$\mathcal{M}_I^\omega(T_1; T_2) = \mathcal{M}_I^\omega(T_2)\mathcal{M}_I^\omega(T_1)$$

我们可以分别分析 $\mathcal{M}_I^\omega(y := 1)(\sigma)$ 和 $\mathcal{M}_I^\omega(\text{while } (x > 0) \text{ do } y := y * x; x := x - 1 \text{ od})(\sigma)$ 。

设

$$\sigma_1 = \mathcal{M}_I^\omega(y := 1)(\sigma) = \sigma[y/1]$$

只需证明 $\mathcal{M}_I^\omega(\text{while } (x > 0) \text{ do } y := y * x; x := x - 1 \text{ od})(\sigma_1)(y) \sqsubseteq (\sigma)(x)!$ 。

定义 $\Phi: [\Sigma_\omega \rightarrow \Sigma_\omega] \rightarrow [\Sigma_\omega \rightarrow \Sigma_\omega]$ 如下。

$$\Phi(f)(\sigma) = \begin{cases} \omega & \text{若 } \sigma = \omega \\ \text{ite}(I(x > 0)(\sigma), f(\mathcal{M}_I^\omega(y := y * x; x := x - 1)(\sigma)), \sigma) & \text{若 } \sigma \neq \omega \end{cases}$$

则

$$\mathcal{M}_I^\omega(T_{jc})(\sigma) = \mu\Phi(\sigma_1)$$

我们需要证明

$$\mu\Phi(\sigma_1)(y) \sqsubseteq \sigma(x)!$$

定义 $g: [\Sigma_\omega \rightarrow \Sigma_\omega]$ 如下

$$g(\sigma) = \begin{cases} \omega & \text{若 } \sigma = \omega \\ \sigma[x/0][y/\sigma(x)! \cdot \sigma(y)] & \text{若 } \sigma \neq \omega \end{cases}$$

我们证明 $\mu\Phi \sqsubseteq g$ 。首先我们证明 g 是 Φ 的一个不动点，即 $\Phi(g)(\sigma) = g(\sigma)$ ，如下。

(1) 若 $\sigma = \omega$ ，则 $\Phi(g)(\sigma) = \omega = g(\sigma)$ 。

(2) 若 $\sigma \neq \omega$ 且 $\sigma(x) = 0$ ，则 $\Phi(g)(\sigma) = \sigma = \sigma[x/0][y/\sigma(x)! \cdot \sigma(y)] = g(\sigma)$ 。

(3) 若 $\sigma \neq \omega$ 且 $\sigma(x) > 0$ ，则

$$\begin{aligned} & \Phi(g)(\sigma) \\ &= g(\mathcal{M}_I^\omega(y := y * x; x := x - 1)(\sigma)) \\ &= g(\mathcal{M}_I^\omega(x := x - 1)\mathcal{M}_I^\omega(y := y * x)(\sigma)) \\ &= g(\mathcal{M}_I^\omega(x := x - 1)(\sigma[y/\sigma(y) \cdot \sigma(x)])) \\ &= g(\sigma[y/\sigma(y) \cdot \sigma(x)][x/\sigma(x) - 1]) \end{aligned}$$

设 $\sigma_2 = \sigma[y/\sigma(y) \cdot \sigma(x)][x/\sigma(x) - 1]$ 。则 $g(\sigma_2) = \sigma_2[x/0][y/\sigma_2(x)! \cdot \sigma_2(y)]$ 。

需要证明 $\sigma_2[x/0][y/\sigma_2(x)! \cdot \sigma_2(y)] = \sigma[x/0][y/\sigma(x)! \cdot \sigma(y)]$ 。

需要证明对任意 z ， $\sigma_2[x/0][y/\sigma_2(x)! \cdot \sigma_2(y)](z) = \sigma[x/0][y/\sigma(x)! \cdot \sigma(y)](z)$ 。

若 z 不是 x, y 则两边都等于 $\sigma(z)$ 。若 z 是 x 则两边都等于 0。若 z 是 y 则两边都等于 $\sigma(x)! \cdot \sigma(y)$ 。因此等式成立。

因此 g 是 Φ 的一个不动点。因此

$$\mu\Phi \sqsubseteq g$$

因此

$$\mu\Phi(\sigma_1)(y) \sqsubseteq g(\sigma_1)(y) = \sigma_1(x)! \cdot \sigma_1(y) = \sigma(x)!$$

Example 1.12 设 T 为

$z = 1; \text{while}(x \neq y) \text{do } z := (y + 1) * (y + 2) * z; y := y + 2 \text{od}$

证明若 $M_I(T)(\sigma)$ 有定义, 则 $M_I(T)(\sigma)(z) = \sigma(x)!/\sigma(y)!$ 。

可定义

$$g(\sigma) = \begin{cases} \sigma[y/\sigma(x)][z/(\sigma(x)!/\sigma(y)!)] & \text{若 } \sigma(x) \geq \sigma(y) \\ & \text{且 } \sigma(x) - \sigma(y) \text{ 为偶数} \\ \omega & \text{否则} \end{cases}$$

或

$$g(\sigma) = \begin{cases} \sigma[y/\sigma(x)][z/(\sigma(x)!/\sigma(y)!)] & \text{若 } \sigma(x) \geq \sigma(y) \\ \omega & \text{否则} \end{cases}$$

在以上证明中, 最困难的部分是循环语句的语义。在证明中定义了一个函数 g 来对应循环语句的语义, 以证明语句初始时的状态和结束时的状态的关系。以下我们定义一个方法直接证明循环语句初始时的状态和结束时的状态的关系。设 $\varphi: \Sigma^2 \rightarrow \text{BOOL}$ 为谓词。

设 $S = \text{while}(e) \text{do } S_1 \text{od}$ 。

$$\forall \sigma \in \Sigma, M_I(S)(\sigma) \downarrow \Rightarrow \varphi(\sigma, M_I(S)(\sigma))$$

若

$$\begin{cases} \forall \sigma \in \Sigma, I(\neg e)(\sigma) \Rightarrow \varphi(\sigma, \sigma) \\ \forall \sigma, \sigma' \in \Sigma, I(e)(\sigma) \wedge M_I(S_1)(\sigma) \downarrow \wedge \varphi(M_I(S_1)(\sigma), \sigma') \Rightarrow \varphi(\sigma, \sigma') \end{cases}$$

证明: 将 φ 看作它的 ω - 扩展, 则 $\forall \sigma \in \Sigma, M_I(S)(\sigma) \downarrow \Rightarrow \varphi(\sigma, M_I(S)(\sigma))$ 即

$$\forall \sigma \in \Sigma, \varphi(\sigma, M_I^\omega(S)(\sigma)) \sqsubseteq \text{true}$$

定义 Φ 如下:

$$\Phi(f)(\sigma) = \begin{cases} \omega & \text{若 } \sigma = \omega \\ \text{ite}(I(e)(\sigma), f(M_I^\omega(S_1)(\sigma)), \sigma) & \text{若 } \sigma \neq \omega \end{cases}$$

则需证 $\forall \sigma \in \Sigma, \varphi(\sigma, \mu\Phi(\sigma)) \sqsubseteq \text{true}$ 。

用不动点归纳法。首先 $\forall \sigma \in \Sigma, \varphi(\sigma, \perp(\sigma)) \sqsubseteq \text{true}$ 成立。剩下的就是要证对任意给定 $f: [\Sigma_\omega \rightarrow \Sigma_\omega]$, $\forall \sigma \in \Sigma, \varphi(\sigma, f(\sigma)) \sqsubseteq \text{true}$ 则 $\forall \sigma' \in \Sigma, \varphi(\sigma', \Phi(f)(\sigma')) \sqsubseteq \text{true}$ 。即要证对任意给定 σ' ,

$$\varphi(\sigma', \text{ite}(I(e)(\sigma'), f(M_I^\omega(S_1)(\sigma')), \sigma')) \sqsubseteq \text{true}$$

若 $I(e)(\sigma') = \text{false}$, 则 $\varphi(\sigma', \text{ite}(I(e)(\sigma'), f(M_I^\omega(S_1)(\sigma')), \sigma')) = \varphi(\sigma', \sigma')$ 。根据前提条件 $\varphi(\sigma', \sigma') \sqsubseteq \text{true}$ 成立。

若 $I(e)(\sigma') = \text{true}$, 则 $\varphi(\sigma', \text{ite}(I(e)(\sigma'), f(M_I^\omega(S_1)(\sigma')), \sigma')) = \varphi(\sigma', f(M_I^\omega(S_1)(\sigma')))$ 。分三种情况。

- 若 $f(M_I^\omega(S_1)(\sigma')) = \omega$, 则 $\varphi(\sigma', f(M_I^\omega(S_1)(\sigma')) \sqsubseteq true$ 成立。
- 若 $f(M_I^\omega(S_1)(\sigma')) \neq \omega$ 而 $M_I^\omega(S_1)(\sigma') = \omega$, 由于 f 的单调性, $f(M_I^\omega(S_1)(\sigma')) = f(\sigma')$ 。根据归纳假设 $\varphi(\sigma', f(\sigma')) \sqsubseteq true$ 成立。
- 若 $f(M_I^\omega(S_1)(\sigma')) \neq \omega$ 且 $M_I^\omega(S_1)(\sigma') \neq \omega$, 根据归纳假设 $\varphi(M_I^\omega(S_1)(\sigma'), f(M_I^\omega(S_1)(\sigma'))) \sqsubseteq true$ 成立。由于 $M_I^\omega(S_1)(\sigma') \neq \omega$ 且 $f(M_I^\omega(S_1)(\sigma')) \neq \omega$, $\varphi(M_I^\omega(S_1)(\sigma'), f(M_I^\omega(S_1)(\sigma'))) = true$ 。根据前提假设 $\varphi(\sigma', f(M_I^\omega(S_1)(\sigma'))) = true$ 成立。

Example 1.13 证明若 $M_I(T_{jc})$ 有定义, 则 $M_I(T_{jc})(\sigma)(y) = \sigma(x)!$ 。

证明: 只需证明 $\mathcal{M}_I^\omega(T_{jc})(\sigma)(y) \sqsubseteq (\sigma)(x)!$ 。由于有 $\mathcal{M}_I^\omega(T_1; T_2) = \mathcal{M}_I^\omega(T_2)\mathcal{M}_I^\omega(T_1)$ 。我们可以分别分析 $\mathcal{M}_I^\omega(y := 1)(\sigma)$ 和 $\mathcal{M}_I^\omega(\text{while } (x > 0) \text{ do } y := y * x; x := x - 1 \text{ od})(\sigma)$ 。设

$$\sigma_1 = \mathcal{M}_I^\omega(y := 1)(\sigma) = \sigma[y/1]$$

只需证明 $\mathcal{M}_I^\omega(\text{while } (x > 0) \text{ do } y := y * x; x := x - 1 \text{ od})(\sigma_1)(y) \sqsubseteq (\sigma)(x)!$ 。设 S_1 为

$$y := y * x; x := x - 1$$

定义

$$\varphi(\sigma, \sigma') = true \text{ 当且仅当 } \sigma'(y) = \sigma(x)! \cdot \sigma(y).$$

若 $I(x > 0)(\sigma) = false$ 则 $\varphi(\sigma, \sigma) = true$ 当且仅当 $\sigma(y) = \sigma(x)! \cdot \sigma(y)$, 由于 $x = 0$, $\varphi(\sigma, \sigma) = true$ 成立。

若 $I(x > 0)(\sigma) = true$, $M_I(S_1)(\sigma) \downarrow$ 且 $\varphi(M_I(S_1)(\sigma), \sigma') = true$, 由于 $M_I(S_1)(\sigma) = \sigma[y/\sigma(y) \cdot \sigma(x)][x/\sigma(x) - 1]$, $\sigma'(y) = \sigma(y) \cdot \sigma(x) \cdot (\sigma(x) - 1)! = \sigma(y) \cdot \sigma(x)!$ 。因此 $\varphi(\sigma, \sigma') = true$ 。

因此 $\forall \sigma \in \Sigma, \varphi(\sigma, M_I(\text{while } (x > 0) \text{ do } S_1 \text{ od})(\sigma)) \sqsubseteq true$ 。

因此 $\varphi(\sigma_1, M_I(\text{while } (x > 0) \text{ do } S_1 \text{ od})(\sigma_1)) \sqsubseteq true$ 。

因此若 $M_I(\text{while } (x > 0) \text{ do } S_1 \text{ od})(\sigma_1)$ 有定义, 则 $M_I(\text{while } (x > 0) \text{ do } S_1 \text{ od})(\sigma_1)(y) = \sigma_1(x)! \cdot \sigma_1(y) = \sigma(x)!$ 。

§1.3.3 Hoare 逻辑

基于指称语义的程序证明具有一定的复杂性。我们还可以用逻辑的方法来证明程序的性质。Hoare 逻辑就是这样的一个可以用来证明程序的性质的系统。Hoare 逻辑语言建立在谓词逻辑和结构化程序的基础上。设 B 是一个谓词逻辑的基本集。Hoare 公式的定义如下。

$$H ::= \{p\}T\{q\}$$

其中 $p, q \in WFF_B$ 是公式, $T \in \mathcal{L}_2^B$ 是结构化程序。
例如以下是一个 Hoare 公式。

$$\{x > 0\}y := 1; \text{while}(x > 0) \text{do } y := y * x; x := x - 1 \text{od} \{y = x!\}$$

设 I 是基本集 B 的解释。Hoare 公式的语义泛函, 记作 \mathcal{I} , 将每个 Hoare 公式 $\{p\}T\{q\}$ 映射到一个 $\Sigma \rightarrow \text{BOOL}$ 的函数。 $\mathcal{I}(\{p\}T\{q\}) : \Sigma \rightarrow \text{BOOL}$ 定义如下。

$\mathcal{I}(\{p\}T\{q\})(\sigma) = \text{true}$ <p>当且仅当</p> $\mathcal{I}(p)(\sigma) = \text{true} \wedge \mathcal{M}_T(T)(\sigma) \downarrow \Rightarrow \mathcal{I}(q)(\mathcal{M}_T(T)(\sigma)) = \text{true}.$
--

一个公式 $\{p\}T\{q\}$ 在 I 的解释下成立, 记为 $\models_I \{p\}T\{q\}$ 。
若其在任意解释下成立, 记为 $\models \{p\}T\{q\}$ 。若其在满足 W 的解释下成立, 记为 $W \models \{p\}T\{q\}$ 。

§1.3.4 Hoare 演算

逻辑公式是一个描述性质的方法。我们需要在这基础上进行推理。我们有以下公理和推理规则。

- 赋值公理: 设 $p \in WFF_B, x \in V, t \in T_B$ 。

$$\{p_x^t\}x := t\{p\}$$

- 顺序复合规则: 设 $p, q, r \in WFF_B, S_1, S_2 \in \mathcal{L}_2^B$ 。

$$\frac{\{p\}S_1\{r\}, \{r\}S_2\{q\}}{\{p\}S_1; S_2\{q\}}$$

- 条件规则: 设 $p, q \in WFF_B, e \in QFF_B, S_1, S_2 \in \mathcal{L}_2^B$ 。

$$\frac{\{p \wedge e\}S_1\{q\}, \{p \wedge \neg e\}S_2\{q\}}{\{p\} \text{if } e \text{ then } S_1 \text{ else } S_2 \text{ fi} \{q\}}$$

- 循环规则: 设 $p \in WFF_B, e \in QFF_B, S_1 \in \mathcal{L}_2^B$ 。

$$\frac{\{p \wedge e\}S_1\{p\}}{\{p\} \text{while } (e) \text{ do } S_1 \text{ od} \{p \wedge \neg e\}}$$

- 推论规则: 设 $p, q, r, s \in WFF_B, S_1 \in \mathcal{L}_2^B$ 。

$$\frac{p \supset q, \{q\}S_1\{r\}, r \supset s}{\{p\}S_1\{r\}}$$

一个 Hoare 公式 $\{p\}T\{q\}$ 可由以上规则和公理得到, 记作 $\vdash \{p\}T\{q\}$ 。为了推理的方便我们可以使用导出规则。导出规则可以看作是基本规则的组合应用。若用以上规则和公理可以从公式 s_1, s_2, \dots, s_n 推导出 s , 则可以产生以下导出规则。

$$\frac{s_1, s_2, \dots, s_n}{s}$$

以下是几个常用的导出规则。在有些时候可以用来替代原有的赋值公理, 顺序复合规则, 条件规则和循环规则的使用。

- 设 $p, q \in WFF_B, x \in V, t \in T_B$.

$$\frac{p \supset q_x}{\{p\}x := t\{q\}}$$

- 设 $p_0, \dots, p_n \in WFF_B, S_1, \dots, S_n \in \mathcal{L}_2^B, n \geq 2$.

$$\frac{\{p_0\}S_1\{p_1\}, \{p_1\}S_1\{p_2\}, \dots, \{p_{n-1}\}S_n\{p_n\}}{\{p_0\}S_1; S_2; \dots; S_n\{p_n\}}$$

- 设 $p_1, p_2, q \in WFF_B, e \in QFF_B, S_1, S_2 \in \mathcal{L}_2^B$.

$$\frac{\{p_1\}S_1\{q\}, \{p_2\}S_2\{q\}}{\{(p_1 \wedge e) \vee (p_2 \wedge \neg e)\} \text{if } e \text{ then } S_1 \text{ else } S_2 \text{ fi}\{q\}}$$

- 设 $p, q, r \in WFF_B, e \in QFF, S_1 \in \mathcal{L}_2^B$.

$$\frac{p \supset r, \{r \wedge e\}S_1\{r\}, (r \wedge \neg e) \supset q,}{\{p\}\text{while}(e) \text{ do } S_1 \text{ od}\{q\}}$$

由于推导中需要用到 $p \supset r$ 等公式。我们还需要一套推导这些公式的方法。

为了避开这些麻烦，专注于与程序直接相关的性质，我们通常将这些成立的式子归于一个理论，在这理论之下，我们可以将这些当成公理来看待。常用的理论包括自然数的理论，记作 PA 。

例子

给定以下程序，记为 T_1 。

```

 $y_1 := 0; y_2 := 1; y_3 := 1;$ 
while  $y_3 \leq x$  do
   $y_1 := y_1 + 1;$ 
   $y_2 := y_2 + 2;$ 
   $y_3 := y_2 + y_3;$ 
od

```

证明 $PA \vdash \{x = c\}T_1\{y_1 = \sqrt{c}\}$ 。

证明

首先将程序分为四部分，记为 S_1, S_2, S_3, S_4 。
我们有

$$\{x = c\}S_1\{x = c \wedge y_1 = 0\}$$

$$\{x = c \wedge y_1 = 0\}S_2\{x = c \wedge y_1 = 0 \wedge y_2 = 1\}$$

$$\{x = c \wedge y_1 = 0 \wedge y_2 = 1\}S_3\{x = c \wedge y_1 = 0 \wedge y_2 = 1 \wedge y_3 = 1\}$$

因此根据顺序复合规则，有

$$\{x = c\}S_1; S_2; S_3\{x = c \wedge y_1 = 0 \wedge y_2 = 1 \wedge y_3 = 1\}$$

根据顺序复合规则，还需要证明

$$\{x = c \wedge y_1 = 0 \wedge y_2 = 1 \wedge y_3 = 1\}S_4\{y_1^2 \leq c \wedge c < (y_1 + 1)^2\}$$

我们有

$$\{x = c \wedge (y_1 + 1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

$$y_1 := y_1 + 1$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1)^2 \wedge y_2 = 2 * (y_1 - 1) + 1\}$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1)^2 \wedge y_2 = 2 * (y_1 - 1) + 1\}$$

$$y_2 := y_2 + 2$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

$$y_3 := y_2 + y_3$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

因此根据顺序复合规则，有

$$\{x = c \wedge (y_1 + 1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

$$y_1 := y_1 + 1; y_2 := y_2 + 2; y_3 := y_2 + y_3$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

由于

$$y_3 \leq x \wedge x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1$$

$$\rightarrow x = c \wedge (y_1 + 1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1$$

有

$$\{y_3 \leq x \wedge x = c \wedge (y_1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

$$y_1 := y_1 + 1; y_2 := y_2 + 2; y_3 := y_2 + y_3$$

$$\{x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

因此根据循环规则，有

$$\{x = c \wedge (y_1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

S_4

$$\{\neg(y_3 \leq x) \wedge x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1\}$$

因此

$$\{x = c \wedge y_1 = 0 \wedge y_2 = 1 \wedge y_3 = 1\}S_4\{y_1^2 \leq c \wedge c < (y_1 + 1)^2\}$$

例子

给定以下程序，记为 T_2 。

```

 $y_1 := x;$ 
 $y_2 := 1;$ 
while  $y_1 \leq 100 \vee y_2 \neq 1$  do
  if  $y_1 \leq 100$  then
     $y_1 := y_1 + 11;$ 
     $y_2 := y_2 + 1;$ 
  else
     $y_1 := y_1 - 10;$ 
     $y_2 := y_2 - 1;$ 
  fi
od
 $z := y_1 - 10;$ 

```

证明 $PA \vdash \{x \leq 100\}T_2\{z = 91\}$ 。

证明

根据顺序复合规则，只需要证明

$$\{x \leq 100 \wedge y_1 = x \wedge y_2 = 1\}S_1\{y_1 = 101\}$$

有

$$\begin{aligned} & \{y_1 \leq 121 \wedge y_2 \geq 2 \wedge (y_1 \geq 111 \wedge y_2 = 2 \rightarrow y_1 = 111)\} \\ & y_1 := y_1 - 10; y_2 := y_2 - 1 \\ & \{y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101)\} \end{aligned}$$

$$\begin{aligned} & \{y_1 \leq 100 \wedge y_2 \geq 0 \wedge (y_1 \geq 100 \wedge y_2 = 0 \rightarrow y_1 = 100)\} \\ & y_1 := y_1 + 11; y_2 := y_2 + 1 \\ & \{y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101)\} \end{aligned}$$

因此

$$\begin{aligned} & \{y_1 \leq 100 \wedge y_1 \leq 100 \wedge y_2 \geq 0 \wedge (y_1 \geq 100 \wedge y_2 = 0 \rightarrow y_1 = 100) \vee \\ & \neg(y_1 \leq 100) \wedge y_1 \leq 121 \wedge y_2 \geq 2 \wedge (y_1 \geq 111 \wedge y_2 = 2 \rightarrow y_1 = 111)\} \\ & \text{if } y_1 \leq 100 \text{ then } y_1 := y_1 + 11; y_2 := y_2 + 1; \\ & \text{else } y_1 := y_1 - 10; y_2 := y_2 - 1; \text{ fi} \\ & \{y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101)\} \end{aligned}$$

因此

$$\begin{aligned} & \{y_1 \leq 100 \wedge y_2 \geq 0 \wedge (y_1 \geq 100 \wedge y_2 = 0 \rightarrow y_1 = 100) \vee \\ & y_1 > 100 \wedge y_1 \leq 121 \wedge y_2 \geq 2 \wedge (y_1 \geq 111 \wedge y_2 = 2 \rightarrow y_1 = 111)\} \\ & \text{if } y_1 \leq 100 \text{ then } y_1 := y_1 + 11; y_2 := y_2 + 1; \\ & \text{else } y_1 := y_1 - 10; y_2 := y_2 - 1; \text{ fi} \\ & \{y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101)\} \end{aligned}$$

又有

$$(y_1 \leq 100 \vee y_2 \neq 1) \wedge y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101)$$

→

$$y_1 \leq 100 \wedge y_2 \geq 0 \wedge (y_1 \geq 100 \wedge y_2 = 0 \rightarrow y_1 = 100) \vee$$

$$y_1 > 100 \wedge y_1 \leq 121 \wedge y_2 \geq 2 \wedge (y_1 \geq 111 \wedge y_2 = 2 \rightarrow y_1 = 111)$$

因此

$$\{ (y_1 \leq 100 \vee y_2 \neq 1) \wedge y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101) \}$$

if $y_1 \leq 100$ then $y_1 := y_1 + 11$; $y_2 := y_2 + 1$;

else $y_1 := y_1 - 10$; $y_2 := y_2 - 1$; fi

$$\{ y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101) \}$$

因此，根据循环规则

$$\{ y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101) \}$$

S_1

$$\{ \neg(y_1 \leq 100 \vee y_2 \neq 1) \wedge y_1 \leq 111 \wedge y_2 \geq 1 \wedge (y_1 \geq 101 \wedge y_2 = 1 \rightarrow y_1 = 101) \}$$

根据推论规则，

$$\{ x \leq 100 \wedge y_1 = x \wedge y_2 = 1 \} S_1 \{ y_1 = 101 \}$$

应用 Hoare 演算证明程序的性质的难点在于循环不变式。

由于这是难点，我们可以考虑从不同侧面证明循环语句的正确性。

设 S 为

$$\text{while } (e) \text{ do } S_1 \text{ od}$$

的初始状态为 σ 且语句经过 m 次执行后终止。

考虑

$$\varphi(\sigma, \sigma')$$

为前后状态之间的关系。

在 e 不成立时 $\varphi(M_I(S_1)^m(\sigma), M_I(S_1)^m(\sigma))$ 成立，

而 e 成立时则

$$\varphi(M_I(S_1)^{m-1}(\sigma), M_I(S_1)^m(\sigma)), \varphi(M_I(S_1)^{m-2}(\sigma), M_I(S_1)^m(\sigma)),$$

直至 $\varphi(\sigma, M_I(S_1)^m(\sigma))$ 必须成立。

我们可以用公式 r 表示 $\varphi(\sigma, \sigma')$ 。

给定 σ, σ' ， r 和 φ 有以下关系。

$$\varphi(\sigma, \sigma') = I(r)(\sigma[x'_1/\sigma'(x_1)] \cdots [x'_n/\sigma'(x_n)])$$

比如 $\sigma'(y) = \sigma(x)! \cdot \sigma(y)$ 用 $y' = x! \cdot y$ 来表示。

设 $p, q, r \in WFF_B, e \in QFF, S_1 \in \mathcal{L}_2^B$ 。则

$$\frac{\neg e \supset r_{x'_1, \dots, x'_n}^{x_1, \dots, x_n}, \{ \neg r \wedge e \} S_1 \{ \neg r \}, (p \wedge r) \supset q_{x_1, \dots, x_n}^{x'_1, \dots, x'_n}}{\{ p \} \text{while } (e) \text{ do } S_1 \text{ od } \{ q \}}$$

这规则中 p, q 只用 x, y 这样的变量，而 r 中用 x, y, x', y' 来表示循环语句开始和结束时的状态的关系。

例子

证明

$$\{x = a \wedge y = 1\}$$

$$\text{while } x > 0 \text{ do } y := y * x; x := x - 1 \text{ od}$$

$$\{y = a!\}$$

设 r 为 $y' = x! * y$.

根据规则需证

$$\neg x > 0 \rightarrow y = x! * y$$

$$\{x > 0 \wedge \neg y' = x! * y\} y := y * x; x := x - 1 \{ \neg y' = x! * y \}$$

$$(x = a \wedge y = 1 \wedge y' = x! * y) \rightarrow y' = a!$$

§1.3.5 可靠与完备

推理系统的可靠指的是推导出的公式确实是成立的, 即若 $W \vdash \{p\}T\{q\}$ 则 $W \models \{p\}T\{q\}$ 。首先对于公理 $\{p_x^t\}x := t\{p\}$, 即 $\vdash \{p_x^t\}x := t\{p\}$ 。我们必须证明 $\models \{p_x^t\}x := t\{p\}$, 即对任意 I , $\models_I \{p_x^t\}x := t\{p\}$ 。我们必须证明对任意 σ , $I(\{p_x^t\}x := t\{p\})(\sigma) = true$ 。根据定义 $I(\{p_x^t\}x := t\{p\})(\sigma) = true$ 当且仅当

$$I(p_x^t)(\sigma) = true \wedge \mathcal{M}_I(x := t)(\sigma) \downarrow \Rightarrow \mathcal{I}(q)(\mathcal{M}_I(x := t)(\sigma)) = true$$

由于

$$I(p_x^t)(\sigma) = I(p)(\sigma[x/I(t)(\sigma)]) = \mathcal{I}(q)(\mathcal{M}_I(x := t)(\sigma))$$

因此

$$I(p_x^t)(\sigma) = true \wedge \mathcal{M}_I(x := t)(\sigma) \downarrow \Rightarrow \mathcal{I}(q)(\mathcal{M}_I(x := t)(\sigma)) = true$$

因此

$$I(\{p_x^t\}x := t\{p\})(\sigma) = true$$

对于顺序复合规则, 我们必须证明 $\models_I \{p\}T_1\{q\}$ 且 $\models_I \{q\}T_2\{r\}$ 则 $\models_I \{p\}T_1; T_2\{r\}$ 。若 $M_I(T_1; T_2)(\sigma)$ 无定义, 则 $I(\{p\}T_1; T_2\{r\})(\sigma) = true$ 。假设 $M_I(T_1; T_2)(\sigma)$ 有定义且 $M_I(T_1; T_2)(\sigma) = \sigma'$ 。需要证明的就是 $I(p)(\sigma)$ 则 $I(r)(\sigma')$ 。 $M_I(T_1; T_2)(\sigma) = \sigma'$ 则 $(T_1; T_2, \sigma) \xrightarrow{*} \sigma'$ 。因此存在 σ'' 使得

$$(T_1; T_2, \sigma) \xrightarrow{*} (T_2, \sigma'') \xrightarrow{*} \sigma'$$

因此

$$(T_1, \sigma) \xrightarrow{*} \sigma'' \text{ 且 } (T_2, \sigma'') \xrightarrow{*} \sigma'$$

根据 $\models_I \{p\}T_1\{q\}$ 和 $\models_I \{q\}T_2\{r\}$, 若 $I(p)(\sigma)$, 则 $I(q)(\sigma'')$, 则 $I(r)(\sigma')$ 。因此

$$I(\{p\}T_1; T_2\{r\})(\sigma) = true$$

对于条件规则，我们必须证明 $\models_I \{p \wedge e\}T_1\{q\}$ 且 $\models_I \{p \wedge \neg e\}T_2\{q\}$ 则 $\models_I \{p\}if\ e\ then\ S_1\ else\ S_2\ fi\{q\}$ 。
若 $M_I(if\ e\ then\ S_1\ else\ S_2\ fi)(\sigma)$ 无定义，则结论成立。
若 $M_I(if\ e\ then\ S_1\ else\ S_2\ fi)(\sigma) = \sigma'$ ，我们分两种情况讨论。若 $I(e)(\sigma)$ 成立则

$$M_I(if\ e\ then\ S_1\ else\ S_2\ fi)(\sigma) = M_I(S_1)(\sigma)$$

根据 $\models_I \{p \wedge e\}T_1\{q\}$ ，若 $I(p)(\sigma)$ 则 $I(q)(M_I(S_1)(\sigma))$ 成立。因此

$$\models_I \{p\}if\ e\ then\ S_1\ else\ S_2\ fi\{q\}$$

同理，若 $I(\neg e)(\sigma)$ 成立，根据 $\models_I \{p \wedge \neg e\}T_2\{q\}$ ，以上命题也成立。

对于循环规则，我们必须证明 $\models_I \{p \wedge e\}T_1\{p\}$ 则 $\models_I \{p\}while\ (e)\ do\ S_1\ od\{p \wedge \neg e\}$ 。若 $M_I(while\ (e)\ do\ S_1\ od)(\sigma)$ 无定义，则结论成立。若 $M_I(while\ (e)\ do\ S_1\ od)(\sigma) = \sigma'$ ，则 $(while\ (e)\ do\ S_1\ od, \sigma) \xrightarrow{k} \sigma'$ 。我们用归纳法证明对任意 σ ， $I(p)(\sigma)$ 则 $I(p \wedge \neg e)(\sigma)$ 。若 $k = 1$ ，则 $\sigma' = \sigma$ ， $I(e)(\sigma) = false$ 。因此 $I(p)(\sigma) \Rightarrow I(p \wedge \neg e)(\sigma')$ 。若 $k > 1$ ，假设对任意 σ ， $i < k$ ， $(while\ (e)\ do\ S_1\ od, \sigma) \xrightarrow{i} \sigma'$ 。则 $I(p)(\sigma) \Rightarrow I(p \wedge \neg e)(\sigma')$ 。由于 $(while\ (e)\ do\ S_1\ od, \sigma) \xrightarrow{k} \sigma'$ 且 $k > 1$ ，则

$$I(e)(\sigma) \text{ 且 } (while\ (e)\ do\ S_1\ od, \sigma) \Rightarrow (S_1; while\ (e)\ do\ S_1\ od, \sigma) \xrightarrow{k-1} \sigma'。$$

存在 σ'' 使得

$$(S_1, \sigma) \xrightarrow{k_1} \sigma'' \text{ 且 } (while\ (e)\ do\ S_1\ od, \sigma'') \xrightarrow{k_2} \sigma'$$

根据 $\models_I \{p \wedge e\}T_1\{p\}$ ，由于已有 $I(e)(\sigma)$ ，若 $I(p)(\sigma)$ ，则 $I(p)(\sigma'')$ 。根据归纳假设有 $I(p)(\sigma'')$ 则 $I(p \wedge \neg e)(\sigma')$ 。因此

$$\models_I \{p\}while\ (e)\ do\ S_1\ od\{p \wedge \neg e\}$$

以上证明了系统的可靠性。

相对的完备性

对于完备性，我们只能证明相对的完备性。首先我们需要具有足够强表达能力的 I 。 I 的表达能力强 enough 的含义就是对任意的程序 T 和公式 q ，存在公式 r 使得 $I(r)$ 为 T 和 $I(q)$ 的最弱宽松前断言。在此情况下，我们称谓词 $I(r)$ 是可表达的。

对于给定的 I 且 I 的表达能力强 enough，

我们证明 $\models_I \{p\}T\{q\}$ 则 $th(I) \vdash \{p\}T\{q\}$ 。证明基于 T 的结构。我们首先证明以下事实。

$\models_I \{p\}S_1; S_2\{q\}$ 且 $I(r)$ 为 S_2 和 $I(q)$ 的最弱宽松前断言, 则 $\models_I \{p\}S_1\{r\}$ 且 $\models_I \{r\}S_2\{q\}$ $\models_I \{p\}$ if (e) then T_1 else T_2 fi $\{q\}$, 则 $\models_I \{p \wedge e\}T_1\{q\}$ 且 $\models_I \{p \wedge \neg e\}T_2\{q\}$ $\models_I \{p\}$ while (e) do T_1 od $\{q\}$, 则 $\models_I \{p\}$ if (e) then T_1 ;while (e) do T_1 od else $x := x$ fi $\{q\}$
--

若 $\models_I \{p\}x := t\{q\}$ 则根据语义有 $\models_I p \supset q_x^t$ 。因此 $th(I) \vdash p \supset q_x^t$ 。根据赋值公理和推论规则可以得出

$$th(I) \vdash \{p\}T\{q\}$$

若 $\models_I \{p\}S_1; S_2\{q\}$ 。设 $I(r)$ 为 S_2 和 $I(q)$ 的最弱宽松前断言。由于 $\models_I \{p\}S_1\{r\}$ 且 $\models_I \{r\}S_2\{q\}$ ，则 $th(I) \vdash \{p\}S_1\{r\}$ 且 $th(I) \vdash \{r\}S_2\{q\}$ 。根据顺序复合规则可以得出

$$th(I) \vdash \{p\}S_1; S_2\{q\}$$

若 $\models_I \{p\}$ if (e) then T_1 else T_2 fi $\{q\}$ 。由于 $\models_I \{p \wedge e\}T_1\{q\}$ 且 $\models_I \{p \wedge \neg e\}T_2\{q\}$ 。则 $th(I) \vdash \{p \wedge e\}T_1\{q\}$ 且 $th(I) \vdash \{p \wedge \neg e\}T_2\{q\}$ 。根据条件规则可以得出

$$th(I) \vdash \{p\}$$
if (e) then T_1 else T_2 fi $\{q\}$

若 $\models_I \{p\}$ while (e) do T_1 od $\{q\}$ 。设 $I(r)$ 为 while (e) do T_1 od 和 $I(q)$ 的最弱宽松前断言。则

$$\begin{aligned} &\models_I p \supset r \text{ 且} \\ &\models_I \{r\}$$
if (e) then T_1 ; while (e) do T_1 od else $x := x$ fi $\{q\}$

则

$$\begin{aligned} &\models_I \{r \wedge e\}T_1;\text{while (e) do } T_1 \text{ od}\{q\} \text{ 且} \\ &\models_I \{r \wedge \neg e\}x := x\{q\}。 \end{aligned}$$

则

$$\models_I \{r \wedge e\}T_1\{r\} \text{ 且 } \models_I (r \wedge \neg e) \supset q。$$

根据循环规则和推论规则可以得出 $th(I) \vdash \{p\}T\{q\}$ 。以上四种情况证明了系统的相对完备性。

§1.3.6 扩展的 Hoare 逻辑

Hoare 逻辑只能证明程序的部分正确，不能证明程序的终止性和完全正确。为了证明程序的终止性和完全正确，有必要对 Hoare 逻辑进行扩展。设 B 是一个谓词逻辑的基本集。扩展的 Hoare 公式的定义如下。

$$H ::= [p]T[q]$$

其中 $p, q \in WFF_B$ 是公式， $T \in \mathcal{L}_2^B$ 是结构化程序。例如以下是一个扩展的 Hoare 公式。

$$[x > 0]y := 1; \text{while}(x > 0)\text{do } y := y * x; x := x - 1 \text{od} [y = x!]$$

设 I 是基本集 B 的解释。扩展的 Hoare 公式的语义泛函，记作 \mathcal{I} ，将每个扩展的 Hoare 公式 $[p]T[q]$ 映射到一个 $\Sigma \rightarrow \text{BOOL}$ 的函数。 $\mathcal{I}([p]T[q]) : \Sigma \rightarrow \text{BOOL}$ 定义如下。

$$\begin{aligned} \mathcal{I}(\{p\}T\{q\})(\sigma) &= \text{true} \\ &\text{当且仅当} \\ \mathcal{I}(p)(\sigma) &= \text{true} \Rightarrow \mathcal{M}_{\mathcal{I}}(T)(\sigma) \downarrow \wedge \mathcal{I}(q)(\mathcal{M}_{\mathcal{I}}(T)(\sigma)) = \text{true}. \end{aligned}$$

结构化程序的是否终止取决于循环语句。为了保证循环语句的终止，我们引入 WFS 集合的应用。首先我们必须保证能够用公式定义需要的 WFS 集合。这样，对 B 和 $I = (D, I_0)$ 有以下要求

- B 包含一个二元谓词符号，记作 \leq 。
- D 包含一个子集 W 且 $(W, I_0(\leq))$ 为一 WFS。
- 存在 $w \in \text{WFF}_B$ 且 w 包含不多于一个变量，记为 x ，满足 $W = \{\sigma(x) \mid \sigma \in \Sigma, \mathcal{I}(w)(\sigma) = \text{true}\} \subseteq D$ 。

这样循环规则可以写为

$$\frac{(p \wedge e) \supset w[x/t], [p \wedge e \wedge t = y]S_1[p \wedge t < y]}{[p]\text{while}(e) \text{ do } S_1 \text{ od}[p \wedge \neg e]}$$

对于赋值，顺序复合和条件语句的规则，可以照 Hoare 演算中的规则把 $\{p\}T\{q\}$ 替换成 $[p]T[q]$ 即可。

例子

Example 1.14 证明 $PA \vdash [x = c]T_1[y_1 = \sqrt{c}]$ 。

需证明

$$[x = c \wedge (y_1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1]$$

S_4

$$\neg(y_3 \leq x) \wedge x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1$$

取 $w = \text{true}$ ， $t = x + 1 - y_3$

需证明

$$[x = c \wedge (y_1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge (y_3 \leq x) \wedge x + 1 - y_3 = a]$$

$$y_1 := y_1 + 1; y_2 := y_2 + 2; y_3 := y_2 + y_3$$

$$x = c \wedge y_1^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge x + 1 - y_3 < a$$

需证明

$$x = c \wedge (y_1)^2 \leq x \wedge y_3 = (y_1 + 1)^2 \wedge y_2 = 2 * y_1 + 1 \wedge (y_3 \leq x) \wedge x + 1 - y_3 = a$$

\rightarrow

$$x = c \wedge (y_1 + 1)^2 \leq x \wedge y_3 + (y_2 + 2) = (y_1 + 1 + 1)^2 \wedge y_2 = 2 * (y_1 + 1) + 1 \wedge$$

$$x + 1 - (y_3 + (y_2 + 1)) < a$$

§1.3.7 工具

程序验证是很烦琐的事情。

需要工具的帮助。

以 XYZ/VERI-II 为例，将 XYZ/SE 格式的循环程序、前断言、后断言以及循环不变式作为输入，我们可以得到循环程序正确的条件。

在此基础上可以做些简化，其余部分的验证工作可由人工完成或交给专门的定理证明工具协助验证。

例子

带有前断言、后断言和循环不变式的 XYZ/SE 格式的程序：

```
{x=c}
%PROC w1(%INP/x:INT;%IOP/y1:INT)==
%LOC [y2,y3:INT]
%STM [
  LB=START => $Oy1=0 ∧ $Oy2=1 ∧ $Oy3=1 ∧ $OLB=l2;
  *[
    LB=l2 ∧ (le(y3,x)) => ($OLB=l3 | $OLB=END)
    LB=l3 => $Oy1=+(y1,1) ∧ $Oy2 = +(y2,2) ∧ $OLB=l4;
    LB=l4 => $Oy3=+(y2,y3) ∧ $OLB=l2;
    {x = c ∧ le(*(y1, y1), x) ∧ y3 = *(+(y1, 1), +(y1, 1)) ∧ y2 = +(*(2, y1), 1)}
  ]
]
```

为阅读方便，我们 $le, lt, +, *$ 等表示为我们熟悉的写法。由这个程序，我们可以用 XYZ/VERI-II 获取以下验证条件

$$y3 \leq x \wedge x = c \wedge y1 * y1 \leq x \wedge y3 = (y1 + 1) * (y1 + 1) \wedge y2 = 2 * y1 + 1$$

→

$$x = c \wedge (y1 + 1) * (y1 + 1) \leq x$$

$$\wedge (y2 + 2) + y3 = ((y1 + 1) + 1) * ((y1 + 1) + 1)$$

$$\wedge y2 + 2 = 2 * (y1 + 1) + 1$$

$$\neg y3 \leq x \wedge x = c \wedge y1 * y1 \leq x \wedge y3 = (y1 + 1) * (y1 + 1) \wedge y2 = 2 * y1 + 1$$

→

$$y1 * y1 \leq c \wedge c < (y1 + 1) * (y1 + 1)$$

$$x = c$$

$$\rightarrow x = c \wedge 0 * 0 \leq x \wedge 1 = (0 + 1) * (0 + 1) \wedge 1 = 2 * 0 + 1$$

通过自动化简得到以下条件：

$(y1 + 1) * (y1 + 1) \leq c, y1 * y1 \leq c \rightarrow 2 + (1 + y1 * 2) = 1 + 2 * (y1 + 1)$
$y1 * y1 \leq c \rightarrow c < (y1 + 1) * (y1 + 1), (y1 + 1) * (y1 + 1) \leq c$
$T \rightarrow 1 = 1 + 2 * 0$
$(y1 + 1) * (y1 + 1) \leq c, y1 * y1 \leq c$ $\rightarrow (2 + (1 + y1 * 2)) + (y1 + 1) * (y1 + 1) = (1 + (y1 + 1)) * (1 + (y1 + 1))$
$T \rightarrow 0 * 0 \leq c$
$T \rightarrow 1 = (1 + 0) * (1 + 0)$

习题

设

$$B = (\{i, j, k, l, x, y, a, b\}, \{0, 1, 2, 3, \dots, +, -, *\}, \{<, =, >\})$$

给定以下 \mathcal{L}_1^B 中的程序 T :

```

-----
i:=1; j:=0; k:=0; l:=1;
while  $\neg(x = y)$  do
  if  $x > y$  then
    x:=x-y; i:=i-k; j:=j-1;
  else
    y:=y-x; k:=k-i; l:=l-j;
  fi
od
-----

```

给定 I 为 B 在整数上的正常解释。证明以下命题成

立:

- | |
|---|
| <p>(1) $\vdash_I \{x = a \wedge y = b \wedge a \geq 0 \wedge b \geq 0\} \text{ T } \{x = \text{gcd}(a, b) \wedge x = i * a + j * b\}$.</p> <p>(2) $\vdash_I [x = a \wedge y = b \wedge a > 0 \wedge b > 0] \text{ T } [x = \text{gcd}(a, b) \wedge x = i * a + j * b]$</p> |
|---|

参考文献

- [1] Zohar Manna and Amir Pnueli. How to cook a temporal proof system for your pet language. The 10th ACM Symposium on Principles of Programming Languages 141-154. 1983.
- [2] Doron A. Peled. Software Reliability Methods. Springer-Verlag. 2001.
- [3] Jacques Loeckx and Kurt Sieber. The foundation of program verification. John Wiley & Sons Ltd., 1984.
- [4] Nissim Francez. Program verification. Addison-Wesley Publishing Company Inc., 1992.
- [5] Wenhui Zhang. Verification of XYZ/SE programs. Chinese Journal of Advanced Software Research 2(4):364-373, 1995.