中国科学院研究生院课程讲稿

§4 推理验证方法

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。

§4.1 卫式迁移系统的推理

本节介绍以卫式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,V) 上的卫式迁移系统。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：

$(T,\Theta)\models_I \varphi$
当且仅当
对所有 I 解释下的 (T,Θ) 的运行 π, $\pi\models_I \varphi$

Definition 4.1 迁移公式的定义如下。

$$Q ::= \{p\}t\{q\}$$

其中 $p,q\in WFF_B$ 是公式， $t\in T$ 是迁移。

迁移公式的语义泛函，记作 I ，将每个迁移公式 $\{p\}t\{q\}$ 映射到一个 $\Sigma\to Bool$ 的函数。$I(\{p\}t\{q\}):\Sigma\to Bool$ 定义如下。

$$\begin{array}{|c|}\hline \mathcal{I}(\{p\}t\{q\})(\sigma)=true \\ \text{当且仅当} \\ I(p)(\sigma)=true\wedge\sigma\xrightarrow{t}\sigma'\to I(q)(\sigma')=true \\ \hline\end{array}$$

一个公式 $\{p\}t\{q\}$ 在 I 的解释下成立，即对任意 σ ， $\mathcal{I}(\{p\}t\{q\})(\sigma)=true$ ，记作 $\models_{\mathcal{I}}\{p\}t\{q\}$ 。 定义 $\models_I\{\varphi\}T\{\psi\}$ 当且仅当对所有 $t\in T$ ， $\models_I\{\varphi\}t\{\psi\}$ 成立。

§4.1.1 迁移系统的推理规则

给定 (B,V) 上的卫式迁移系统 (T,Θ) 。设 t 为 $p\longrightarrow(x_1,...,x_n):=(e_1,...,e_n)\in T$ 。定义 $COND_t=p$ 和 $COND_T=\bigvee_{t\in T}COND_t$ 。

- 迁移公理：

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

- Θ 规则：

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

- O 规则：

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee COND_T) \\ \{\varphi\} T \{\psi\}\end{array}}{\varphi \Rightarrow O\psi}$$

- 推断规则：

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

§4.1.2 导出规则

由以上规则和一阶线性时序逻辑的推理可以导出以下规则：

- 迁移规则：

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

- R 规则：

$$\frac{\begin{array}{l}\zeta \Rightarrow \varphi' \\ \{\varphi' \wedge \neg\psi\} T \{\varphi'\} \\ \varphi' \Rightarrow \varphi\end{array}}{\zeta \Rightarrow \psi R \varphi}$$

- $\Box$ 规则：

$$\frac{\begin{array}{l}\zeta \Rightarrow \varphi' \\ \{\varphi'\} T \{\varphi'\} \\ \varphi' \Rightarrow \varphi\end{array}}{\zeta \Rightarrow \Box\varphi}$$

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

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee \zeta) \\ \zeta \Rightarrow (\zeta_0 \wedge w_x^e \wedge (\psi \vee COND_T)) \\ \{\zeta \wedge e = v\} T \{\psi \vee (\zeta \wedge e \sqsubset v)\}\end{array}}{\varphi \Rightarrow \zeta_0 U \psi}$$

- $\Diamond$ 规则：

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee \zeta) \\ \zeta \Rightarrow (w_x^e \wedge (\psi \vee COND_T)) \\ \{\zeta \wedge e = v\} T \{\psi \vee (\zeta \wedge e \sqsubset v)\}\end{array}}{\varphi \Rightarrow \Diamond\psi}$$

§4.1.3 变量 O 规则

用扩展的一阶线性时序逻辑，我们可以将迁移完全用公式刻画。

设 $V = \{x_1, ..., x_n\}$ 。给定一个迁移 $t : p \rightarrow (x_1, ..., x_k) = (e_1, ..., e_k)$ 。定义 $\bar{x}(t) = (x_1, ..., x_k, x_{k+1}, ..., x_n)$ ， $\bar{e}(t) = (e_1, ..., e_k, x_{k+1}, ..., x_n)$ 。我们可以定义以下规则。

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

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，某些逻辑是不可能有完备的推理系统的。比如自然数域上的一阶逻辑是不可能有完备的推理系统的。 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) 。

$$\Phi = \ \Theta \wedge \Box(COND_T \rightarrow \bigvee_{t\in T}(COND_t \wedge O\bar{x}(t) = \bar{e}(t))) \wedge \Box(\neg COND_T \rightarrow O\bar{x}(t) = \bar{x}(t))$$

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 $\models_I \Phi \rightarrow \varphi$ 。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。

§4.2 谓词迁移系统的推理

本节介绍以谓词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 (ρ,Θ) 为 (B,V) 上的谓词迁移系统。给定 B 上的解释 I 。 (ρ,Θ) 满足一阶线性时序逻辑公式 φ ，记作 $(\rho,\Theta)\models_I\varphi$ 。其定义如下：

$(\rho,\Theta)\models_I\varphi$
当且仅当
对所有 I 解释下的 (ρ,Θ) 的运行 π, $\pi\models_I\varphi$

Definition 4.2 迁移公式的定义如下。

$$Q ::= \{p\}\rho\{q\}$$

其中 $p,q\in WFF_B$ 是公式。

迁移公式的语义泛函，记作 I ，将每个迁移公式 $\{p\}\rho\{q\}$ 映射到一个 $\Sigma\to Bool$ 的函数。$I(\{p\}\rho\{q\}):\Sigma\to Bool$ 定义如下。

$$\begin{array}{|c|}\hline \mathcal{I}(\{p\}\rho\{q\})(\sigma)=true \\ \text{当且仅当} \\ I(p)(\sigma)=true\wedge\sigma\to\sigma'\to I(q)(\sigma')=true \\ \hline\end{array}$$

一个公式 $\{p\}\rho\{q\}$ 在 I 的解释下成立，即对任意 σ ， $\mathcal{I}(\{p\}\rho\{q\})(\sigma)=true$ ，记作 $\models_{\mathcal{I}}\{p\}\rho\{q\}$ 。

§4.2.1 迁移系统的推理规则

给定 (B,V) 上的谓词迁移系统 (ρ,Θ) 。设 $V=\{v_1,...,v_n\}$ 。 设 φ,ψ 为 $\Phi(V)$ 上的公式。

- 迁移公理：

$$\{\forall v_1',...,v_n'.(\rho\to\psi^{v_1',...,v_n'}_{v_1,...,v_n})\}\ \rho\ \{\psi\}$$

- Θ 规则：

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

- O 规则：

$$\frac{\begin{array}{l}\varphi\Rightarrow(\psi\vee\exists v_1'...v_n'.\rho)\\ \{\varphi\}\rho\{\psi\}\end{array}}{\varphi\Rightarrow O\psi}$$

- 推断规则：

$$\frac{\varphi'\Rightarrow\varphi\quad\{\varphi\}\rho\{\psi\}\quad\psi\Rightarrow\psi'}{\{\varphi'\}\rho\{\psi'\}}$$

§4.2.2 导出规则

由以上规则和一阶线性时序逻辑的推理可以导出以下规则：

- 迁移规则：

$$\frac{\varphi \wedge \rho \to \psi^{v'_1,\ldots,v'_n}_{v_1,\ldots,v_n}}{\{\varphi\}\,\rho\,\{\psi\}}$$

- R 规则：

$$\frac{\begin{array}{l}\zeta \Rightarrow \varphi' \\ \{\varphi' \wedge \neg\psi\}\rho\{\varphi'\} \\ \varphi' \Rightarrow \varphi\end{array}}{\zeta \Rightarrow \psi R \varphi}$$

- $\Box$ 规则：

$$\frac{\begin{array}{l}\zeta \Rightarrow \varphi' \\ \{\varphi'\}\rho\{\varphi'\} \\ \varphi' \Rightarrow \varphi\end{array}}{\zeta \Rightarrow \Box\varphi}$$

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

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee \zeta) \\ \zeta \Rightarrow (\zeta_0 \wedge w^e_x \wedge (\psi \vee \exists v'_1 \ldots v'_n.\rho)) \\ \{\zeta \wedge e = v\}\rho\{\psi \vee (\zeta \wedge e \sqsubset v)\}\end{array}}{\varphi \Rightarrow \zeta_0 U \psi}$$

- $\Diamond$ 规则：

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee \zeta) \\ \zeta \Rightarrow (w^e_x \wedge (\psi \vee \exists v'_1 \ldots v'_n.\rho)) \\ \{\zeta \wedge e = v\}\rho\{\psi \vee (\zeta \wedge e \sqsubset v)\}\end{array}}{\varphi \Rightarrow \Diamond\psi}$$

§4.3 流程图程序的推理

对于流程图程序，传统的性质包括部分正确和完全正确。这些性质约束的是程序运行终止时程序的变量状态，以及程序是否能够终止。

给定程序 $T \in \mathcal{L}^{(B,V)}_{\hookleftarrow}$ 和 B 的解释 I 。定义程序的语义函数为变量状态到变量状态的偏函数 $\mathcal{M}_I(T) : \Sigma \hookrightarrow \Sigma$ 如下。

$$\mathcal{M}_I(T)(\sigma) = \begin{cases} \sigma' & \text{若 } (beg,\sigma) \xrightarrow{*} (end,\sigma') \text{ 。} \\ \text{无定义} & \text{若不存在 } \sigma' \text{ 使得 } (beg,\sigma) \xrightarrow{*} (end,\sigma') \text{ 。} \end{cases}$$

$\mathcal{M}_I(T)(\sigma)$ 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 $\mathcal{M}_I(T)(\sigma)\downarrow$ 。反之则记作 $\mathcal{M}_I(T)(\sigma)\uparrow$ 。

程序性质

程序是否正确相对于给定的断言而言。其断言的描述用谓词或谓词公式。

Definition 4.3 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 解释之下对于 φ 和 ψ 是部分正确的，若 $\forall\sigma\in\Sigma,\varphi(\sigma)=true\to(\mathcal{M}_I(T)(\sigma)\downarrow\to\psi(\mathcal{M}_I(T)(\sigma)))$ 。

Definition 4.4 设 φ 是个谓词。φ 称为前断言。T 在 I 解释之下对于 φ 是终止的，若 $\forall\sigma\in\Sigma,\varphi(\sigma)=true\to\mathcal{M}_I(T)(\sigma)\downarrow$ 。

Definition 4.5 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 解释之下对于 φ 和 ψ 是完全正确的，若 $\forall\sigma\in\Sigma,\varphi(\sigma)=true\to\mathcal{M}_I(T)(\sigma)\downarrow\wedge\psi(\mathcal{M}_I(T)(\sigma))$ 。

程序对于给定公式的正确性的描述

Definition 4.6 设 T 为程序，p 和 q 是两个公式。T 在 I 解释之下对于 p 和 q 是部分正确的，记作 $\models_I\{p\}T\{q\}$ ，若 T 在 I 解释之下对于 $I(p)$ 和 $I(q)$ 是部分正确的。

$$\models_I\{p\}T\{q\}\text{ 当且仅当 }\forall\sigma\in\Sigma,I(p)(\sigma)=true\to(\mathcal{M}_I(T)(\sigma)\downarrow\to I(q)(\mathcal{M}_I(T)(\sigma)))\text{ 。}$$

Definition 4.7 设 T 为程序，p 是公式。T 在 I 解释之下对于 p 是终止的，记作 $\models_I[p]T[true]$ ，若 T 在 I 解释之下对于 $I(p)$ 是终止的。

$$\models_I[p]T[true]\text{ 当且仅当 }\forall\sigma\in\Sigma,I(p)(\sigma)=true\to\mathcal{M}_I(T)(\sigma)\downarrow\text{ 。}$$

Definition 4.8 设 T 为程序，p 和 q 是两个公式。T 在 I 解释之下对于 p 和 q 是完全正确的，记作 $\models_I[p]T[q]$ ，若 T 在 I 解释之下对于 $I(p)$ 和 $I(q)$ 是完全正确的。

$$\models_I[p]T[q]\text{ 当且仅当 }\forall\sigma\in\Sigma,I(p)(\sigma)=true\to(\mathcal{M}_I(T)(\sigma)\downarrow\wedge I(q)(\mathcal{M}_I(T)(\sigma)))\text{ 。}$$

程序正确性的特点

设 $B=(F,P)$ 和 $I=(NAT,I_0)$ ，其中 $F=\{0,1,+\},P=\{=\}$ ，NAT 为自然数，I_0 将符号 $0,1,+$ 解释为自然数上的 $0,1$ 和加法，将符号 $=$ 解释为自然数上的相等关系。以下集合不是递规可枚举的：

$$PC=\{(\varphi,T,\psi)\mid T\in\mathcal{L}^B_{\hookleftarrow},\varphi,\psi\in WFF_B,\models_I\{\varphi\}T\{\psi\}\}$$
$$TE=\{(\varphi,T)\mid T\in\mathcal{L}^B_{\hookleftarrow},\varphi\in WFF_B,\models_I[\varphi]T[true]\}$$
$$TC=\{(\varphi,T,\psi)\mid T\in\mathcal{L}^B_{\hookleftarrow},\varphi,\psi\in WFF_B,\models_I[\varphi]T[\psi]\}$$

最弱前断言和最强后断言

Definition 4.9 设 T 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： $\varphi(\sigma)=true$ 当且仅当 $(M_I(T)(\sigma)\downarrow\to\psi(M_I(T)(\sigma)))$
φ 是 T 和 ψ 的最弱前断言： $\varphi(\sigma)=true$ 当且仅当 $M_I(T)(\sigma)\downarrow\wedge\psi(M_I(T)(\sigma))$
ψ 是 T 和 φ 的最强后断言： $\psi(\sigma)=true$ 当且仅当存在状态 σ' 使得 $\varphi(\sigma')=true\wedge M_I(T)(\sigma')=\sigma$

我们有以下性质：

若 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 $\varphi' \to \varphi$ 。
若 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 $\varphi' \to \varphi$ 。
若 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 $\psi \to \psi'$ 。

推理证明

对于部分正确而言，若前断言为 φ ，后断言为 ψ ，证明程序满足该对前后断言的一种思路是假设程序的运行为

$$(beg, \sigma_0), (l_1, \sigma_1), ..., (l_{n-1}, \sigma_{n-1}), (end, \sigma_n)$$

然后证明若 $\varphi(\sigma_0)$ 则 $\psi(\sigma_n)$ 。

对于终止而言，一种证明思路是假设程序的运行不终止

$$(beg, \sigma_0), (l_1, \sigma_1), ...,$$

那么就会有某个 $l \in LB$ 在运行中出现无限多次。然后证明若 $\varphi(\sigma_0)$ 成立则 l 在某次出现后，与其所对应的 σ 不满足还能够回去执行定义 l 的指令的条件。

完全正确性可以分成部分正确和终止两方面的证明。如果放在一起证明，一种思路是假设程序的运行为

$$(beg, \sigma_0), (l_1, \sigma_1), ...,$$

那么若 $\varphi(\sigma_0)$ 成立则通过 n 次（ n 与初始变量状态 σ_0 相关）状态变化后， $l_n = end$ 且 $\psi(\sigma_n)$ 成立。

基于路径的推理

以上的的证明思路是基于对程序过程的分析。需要首先确定了程序的运行过程。对于简单的程序，该思路是可行的，但对于复杂的程序，确定程序运行过程有一定的难度。 因而我们需要更好的方法。一个重要的原则是将一个大的问题分解成小的问题。对于流程图程序，我们可以将程序的运行分成不同的片段，由这些片段的正确性推导整个程序的正确性。 程序片段用标号序列刻画。称标号序列为路径。

Definition 4.10 给定程序 T 。标号序列 $(l_0, l_1, ..., l_n)$ 是 T 的路径当且仅当 $n > 0$ 且对所有 $0 \le i < n$ ， l_{i+1} 在 l_i 的定义中出现。

设 $\alpha = (l_0, l_1, ..., l_n)$ 为一路径。路径的语义定义如下。

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

若 $l_0 = beg$ 且 $l_n = end$ 则 $\alpha = (l_0, l_1, ..., l_n)$ 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 $\mathcal{M}_I(\alpha)(\sigma) = \sigma'$ 。类似于程序正确性，我们可以定义路径的部分正确性。

Definition 4.11 设 p 和 q 是两个公式。 α 在 I 解释之下对于 p 和 q 是部分正确的，记作 $\models_I \{p\}\alpha\{q\}$ ，若 $\forall \sigma \in \Sigma, I(p)(\sigma) = true \to (\mathcal{M}_I(\alpha)(\sigma) \downarrow \to I(q)(\mathcal{M}_I(\alpha)(\sigma)))$ 。

Definition 4.12 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

$I(p)(\sigma) = true$ 当且仅当 $\mathcal{M}_I(\alpha)(\sigma) \downarrow \rightarrow I(q)(\mathcal{M}_I(\alpha)(\sigma))$ 。

Definition 4.13 设 $\alpha = (l_0, ..., l_k)$ 为一路径。定义 $wlp(\alpha, q)$ 如下。

$k = 1$	存在 T 中指令 l_0: $(v_1, ..., v_n) := (e_1, ..., e_n)$ goto l_1 则 $wlp(\alpha, q) = q[v_1/e_1] \cdots [v_n/e_n]$.
	存在 T 中指令 l_0: if (b) goto l else goto l' 则 $wlp(\alpha, q) = \begin{cases} b \rightarrow q & \text{若 } l = l_1; \\ \neg b \rightarrow q & \text{若 } l' = l_1 \text{ 。} \end{cases}$
$k > 1$	设 $\alpha_0 = (l_0, l_1)$, $\alpha_1 = (l_1, ..., l_k)$ 。则 $wlp(\alpha, q) = wlp(\alpha_0, wlp(\alpha_1, q))$

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

Definition 4.14 设 $\alpha = (l_0, ..., l_k)$ 为一路径。定义 $vc(p, \alpha, q) = p \rightarrow wlp(\alpha, q)$ 。

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

Definition 4.15 设 T 为程序， C 为标号集合。定义 $\gamma_T(C)$ 如下：
$(l_0, ..., l_k) \in \gamma_T(C)$ 当且仅当 $(l_0, ..., l_k)$ 是 T 的路径， $l_0, l_k \in C$ 且 $l_1, ..., l_{k-1} \notin C$ 。

部分正确性证明

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 $\forall \alpha = (l_0, ..., l_k) \in \gamma_T(C), \models_I vc(q_{l_0}, \alpha, q_{l_k})$ ，则 $\models \{q_{beg}\}T\{q_{end}\}$ 。

终止性的特点

终止性不是一阶性质。以下程序在一阶算术 PA (Peano Arithmetic) 的标准模型下对任意输入能够终止，但在 PA 的非标准模型并不一定终止。

```
beg:   (x) := (0); goto test;
test:  if x = y then goto end else goto loop fi;
loop:  (x) := (x + 1); goto test;
```

终止性证明

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 $\models [q_{beg}]T[true]$:

(a)	$\forall \alpha = (l_0, ..., l_k) \in \gamma_T(C), \models_I vc(q_{l_0}, \alpha, q_{l_k})$
(b)	$\forall \alpha = (l_0, ..., l_k) \in \gamma_T(C), I(q_{l_0})(\sigma) = true \wedge M_I(\alpha)(\sigma) \downarrow \rightarrow g_{l_k}(M_I(\alpha)(\sigma)) \sqsubset q_{l_0}(\sigma)$

基于谓词公式的终止性证明

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 。 $(W \subseteq D, I_0(\sqsubseteq))$ 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 $\models [q_{beg}]T[true]$:

(a)	$\forall \alpha = (l_0, ..., l_k) \in \gamma_T(C), \models_I vc(q_{l_0}, \alpha, q_{l_k})$
(b)	$\forall \alpha = (l_0, ..., l_k) \in \gamma_T(C), \models_I vc(q_{l_0} \wedge t_{l_0} = a, \alpha, t_{l_k} \sqsubset a)$

这个方法相对于前面的方法有一定的局限性，即 WFS 的选择受限于 $W \subseteq D$ 和谓词符号必须在 P 中。

§4.4 结构化循环程序的推理

流程图程序的一个缺点是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。结构化循环程序可以克服这些缺点。

给定程序 $T \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 和 B 的解释 I 。定义程序的语义函数为变量状态到变量状态的偏函数 $\mathcal{M}_I(T) : \Sigma \hookrightarrow \Sigma$ 如下。

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

由于结构化循环程序可以看成是流程图程序的特殊形式。流程图程序的证明方法同样适用于结构化循环程序。另外，由于结构化循环程序具有组合性，我们可以根据指称语义和公理语义证明程序性质。

§4.4.1 指称语义

结构化循环程序具有组合性。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 if e then T_1 else T_2 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 $\mathcal{M}_I(T_1;T_2)(\sigma) = \mathcal{M}_I(T_2)\mathcal{M}_I(T_1)(\sigma)$ 。 由于 $\mathcal{M}_I(T_1)(\sigma)$ 不一定有定义，因此以上写法不一定是良定义的。为绕过这个问题，我们对 $\mathcal{M}$ 的定义域和值域进行扩充。定义 $\Sigma_\omega = \Sigma \cup \{\omega\}$ 。扩充后的语义泛函记为 $\mathcal{M}_I^\omega$ 。给定结构化循环程序 T ， $\mathcal{M}_I^\omega(T)$ 是 Σ_ω 上的函数。

定义 $ite : Bool \times \Sigma_\omega \times \Sigma_\omega \to \Sigma_\omega$ 如下

$$\begin{aligned} ite(true,\sigma,\sigma') &= \sigma \\ ite(false,\sigma,\sigma') &= \sigma' \end{aligned}$$

定义 $\Phi_{e,h} : [\Sigma_\omega \to \Sigma_\omega] \to [\Sigma_\omega \to \Sigma_\omega]$ 如下

$$\Phi_{e,h}(f)(\sigma) = \begin{cases} \omega & \text{若 } \sigma = \omega \\ ite(I(e)(\sigma), f(h(\sigma)), \sigma) & \text{若 } \sigma \neq \omega \end{cases}$$

定义 Σ_ω 上的偏序关系 $\leq$ 如下： $a \leq b$ 当且仅当 $a = \omega$ 。 $(\Sigma_\omega, \leq)$ 是完备偏序。定义 $\mathcal{M}_I^\omega(T) : \Sigma_\omega \to \Sigma_\omega$ 如下。

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

对于任意结构化循环程序 T ， $\mathcal{M}_I^\omega(T)$ 是良定义且连续的。基于 $\mathcal{M}_I^\omega(T)$ ，我们可以定义 $\mathcal{M}_I(T) : \Sigma \hookrightarrow \Sigma$ 如下。

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

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

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

部分正确性

部分正确性 $\{\varphi\}T\{\psi\}$ 表示为 $\varphi(\sigma) \to (\mathcal{M}_I(T)(\sigma) \downarrow\to \psi(\mathcal{M}_I(T)(\sigma)))$ 。 如果能够直接通过计算把 $\mathcal{M}_I(T)$ 表示成一个简单函数，则证明就容易了。通常由于部分正确不要求程序一定终止，我们只要计算满足 $\mathcal{M}_I^\omega(T) \sqsubseteq h$ 的一个 Σ 上的函数 h ，然后证明 $\varphi(\sigma) \to \psi(h(\sigma))$ 。

在这样的以证明中，最困难的部分是循环语句的语义。在证明中通常需要定义一个不动点 g 来对应循环语句的语义，以证明循环语句初始时的状态和结束时的状态的关系。我们可以直接定义一个证明循环语句初始时的状态和结束时的状态的关系的方法。

设 $\varphi : \Sigma^2 \to Bool$ 为谓词。设 T=while e do T_1 od 。若以下命题成立，则 $\forall\sigma \in \Sigma, M_I(T)(\sigma) \downarrow\to \varphi(\sigma, M_I(T)(\sigma))$ 。

$$\boxed{\begin{array}{l} \forall\sigma \in \Sigma, I(\neg e)(\sigma) \to \varphi(\sigma, \sigma) \\ \forall\sigma, \sigma' \in \Sigma, I(e)(\sigma) \wedge M_I(T_1)(\sigma) \downarrow \wedge \varphi(M_I(T_1)(\sigma), \sigma') \to \varphi(\sigma, \sigma') \end{array}}$$

这样我们只计算 $M_I(T_1)(\sigma)$ 而避开了不动点，因此该方法的应用可以简化一些循环语句正确性的证明。

终止性

程序 T 的终止性可以根据已知程序 T' 的终止性来证明。若 $M_I(T')(\sigma)$ 为处处有定义的函数且 $M_I(T)(\sigma) \sqsubseteq M_I(T')(\sigma)$ 则 $M_I(T)(\sigma)$ 处处有定义。

§4.4.2 Hoare 逻辑

除了基于指称语义的程序证明，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循环程序的基础上。给定 (B, V) 。

Definition 4.16 *Hoare 公式的定义如下。*

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

其中 $p, q \in WFF_B$ 是公式， $T \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 是结构化循环程序。

设 I 是基本符号集 B 的解释。Hoare 公式的语义泛函，记作 I ，将每个 Hoare 公式 $\{p\}T\{q\}$ 映射到一个 $\Sigma \to Bool$ 的函数。 $I(\{p\}T\{q\}) : \Sigma \to Bool$ 定义如下。

$$\boxed{\begin{array}{c} \mathcal{I}(\{p\}T\{q\})(\sigma) = true \\ \text{当且仅当} \\ \mathcal{I}(p)(\sigma) = true \wedge \mathcal{M}_{\mathcal{I}}(T)(\sigma) \downarrow\to \mathcal{I}(q)(\mathcal{M}_{\mathcal{I}}(T)(\sigma)) = true. \end{array}}$$

一个公式 $\{p\}T\{q\}$ 在 I 的解释下成立，即对任意 σ ， $\mathcal{I}(\{p\}T\{q\})(\sigma) = true$ ，记为 $\models_{\mathcal{I}} \{p\}T\{q\}$ 。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 $W \models \{p\}T\{q\}$ 。

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。

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

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

(2) 组合规则：设 $p, q, r \in WFF_B, T_1, T_2 \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 。

$$\frac{\{p\}T_1\{r\}, \{r\}T_2\{q\}}{\{p\}T_1; T_2\{q\}}$$

(3) 条件规则：设 $p, q \in WFF_B, e \in QFF_B, T_1, T_2 \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 。

$$\frac{\{p \wedge e\}T_1\{q\}, \{p \wedge \neg e\}T_2\{q\}}{\{p\}\text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}\{q\}}$$

(4) 循环规则：设 $p \in WFF_B, e \in QFF_B, T_1 \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 。

$$\frac{\{p \wedge e\}T_1\{p\}}{\{p\}\text{while } e \text{ do } T_1 \text{ od}\{p \wedge \neg e\}}$$

(5) 推论规则：设 $p, q, r, s \in WFF_B, T \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 。

$$\frac{p \to q, \{q\}T\{r\}, r \to s}{\{p\}T\{s\}}$$

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 $s_1, s_2, ..., s_n$ 推导出 s ，则可以产生以下导出规则。

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

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，组合规则，条件规则和循环规则的使用。

(1') 设 $p, q \in WFF_B, x \in V, t \in T_B$.

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

(2') 设 $p_0, ..., p_n \in WFF_B, T_1, ..., T_n \in \mathcal{L}_{\bigcirc}^{(B,V)}, n \geq 2$.

$$\frac{\{p_0\}T_1\{p_1\}, \{p_1\}T_2\{p_2\}, ..., \{p_{n-1}\}T_n\{p_n\}}{\{p_0\}T_1; T_2; ...; T_n\{p_n\}}$$

(3') 设 $p_1, p_2, q \in WFF_B, e \in QFF_B, T_1, T_2 \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 。

$$\frac{p \to (p_1 \wedge e) \vee (p_2 \wedge \neg e), \{p_1\}T_1\{q\}, \{p_2\}T_2\{q\}}{\{p\}\text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}\{q\}}$$

(4') 设 $p, q, r \in WFF_B, e \in QFF, T_1 \in \mathcal{L}_{\bigcirc}^{(B,V)}$.

$$\frac{p \to r, \{r \wedge e\}T_1\{r\}, (r \wedge \neg e) \to q,}{\{p\}\text{while } e \text{ do } T_1 \text{ od}\{q\}}$$

由于推导中需要用到 $p \to r$ 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理论 PA 等。

推理证明

程序的推理证明在程序的每个语句前后放上合适的断言，应用以上规则证明程序的每个语句对于这些断言都是对的。有些断言是可以从需要证明的前后断言通过简单推导得到的，有些是要自己通过分析程序添加的。

可靠性

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 $W \models \{p\}T\{q\}$ 。给定 I 。记 $th(I)$ 为在 I 的解释下成立的所有公式。我们需要保证

$$\text{若 } th(I) \vdash \{p\}T\{q\} \text{ 则 } \models_I \{p\}T\{q\} \text{ 。}$$

我们有以下性质。

$\models_I \{q_x^t\}x := t\{q\}$
$\models_I \{p\}T_1\{r\}$ 且 $\models_I \{r\}T_2\{q\}$ ，则 $\models_I \{p\}T_1; T_2\{q\}$
$\models_I \{p \wedge e\}T_1\{q\}$ 且 $\models_I \{p \wedge \neg e\}T_2\{q\}$ ，则 $\models_I \{p\}\text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}\{q\}$
$\models_I \{p \wedge e\}T_1\{p\}$ ，则 $\models_I \{p\} \text{ while } e \text{ do } T_1 \text{ od}\{p \wedge \neg e\}$
$\models_I \{q\}T\{r\}$ ， $\models_I p \to q$ 且 $\models_I r \to s$ ，则 $\models_I \{p\}T\{q\}$

这些性质对应于前面的一个公理和 4 个推理规则，保证了推理系统的可靠。

相对的完备性

定义 φ 为 T 和 ψ 的最弱宽松前断言如下：

$$\varphi(\sigma) = true \text{ 当且仅当 } \mathcal{M}_{\mathcal{I}}(T)(\sigma) \downarrow \to \psi(\mathcal{M}_{\mathcal{I}}(T)(\sigma)) = true.$$

我们有以下性质。

$\models_I \{p\}x := t\{q\}$ ，则 $\models_I p \to q_x^t$
$\models_I \{p\}T_1; T_2\{q\}$ 且 $I(r)$ 为 T_2 和 $I(q)$ 的最弱自由前断言， 则 $\models_I \{p\}T_1\{r\}$ 且 $\models_I \{r\}T_2\{q\}$
$\models_I \{p\}\text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}\{q\}$ ，则 $\models_I \{p \wedge e\}T_1\{q\}$ 且 $\models_I \{p \wedge \neg e\}T_2\{q\}$
$\models_I \{p\} \text{ while } e \text{ do } T_1 \text{ od}\{q\}$ ，则 $\models_I \{p\}$ if (e) T_1; while e do T_1 od else $x := x$ fi $\{q\}$

对于完备性，我们需要具有足够强表达能力的 I 。 I 的表达能力足够强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 $I(q)$ 的最弱自由前断言。在此情况下，我们称谓词 $I(r)$ 是可表达的。

给定 I 且假定 I 的表达能力足够强，对 T 做结构归纳，我们有

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

即 Hoare 演算系统具有相对完备性。

循环规则的替代规则

循环规则中的 p 称为循环不变式。应用 Hoare 演算证明程序的性质的难点在于添加合适的循环不变式。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。一个替代循环规则的方法可以是从指称语义证明循环语句的方法中导出。指称语义证明循环语句的方法中的谓词 $\varphi(\sigma,\sigma')$ 就是一种不变式。设 T=while e do T_1 od 且 T 以 σ 为初始状态经过 m 次 T_1 的执行后终止。设 $\sigma' = M_I(T_1)^m(\sigma)$ 。考虑 $\varphi(\sigma,\sigma')$ 为前后状态之间的关系，且：

(1) e 不成立时 $\varphi(M_I(T_1)^m(\sigma), M_I(T_1)^m(\sigma))$ 成立。

(2) e 成立时则 $\varphi(M_I(T_1)^{m-1}(\sigma), M_I(T_1)^m(\sigma))$ ， $\varphi(M_I(T_1)^{m-2}(\sigma), M_I(T_1)^m(\sigma))$ ，直至 $\varphi(\sigma, M_I(T_1)^m(\sigma))$ 成立。

我们用公式 r 表示 $\varphi(\sigma,\sigma')$ 。给定 σ,σ' ， r 和 φ 有以下关系。

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

设 $p,q,r \in WFF_B, e \in QFF, T_1 \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 。则

$$\frac{\neg e \to r^{x_1,\ldots,x_n}_{x_1',\ldots,x_n'}, \{\neg r \wedge e\}T_1\{\neg r\}, (p \wedge r) \to q^{x_1',\ldots,x_n'}_{x_1,\ldots,x_n},}{\{p\}\text{while } e \text{ do } T_1 \text{ od}\{q\}}$$

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。

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。给定 (B,V) 。

Definition 4.17 *Hoare 公式的定义如下。*

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

其中 $p,q \in WFF_B$ 是公式， $T \in \mathcal{L}_{\bigcirc}^{(B,V)}$ 是结构化循环程序。

设 I 是基本符号集 B 的解释。扩展的 Hoare 公式的语义泛函，记作 I ，将每个扩展的 Hoare 公式 $[p]T[q]$ 映射到一个 $\Sigma \to Bool$ 的函数。 $I([p]T[q]) : \Sigma \to Bool$ 定义如下。

$$\begin{array}{|c|}\hline \mathcal{I}([p]T[q])(\sigma) = true \\ \text{当且仅当} \\ \mathcal{I}(p)(\sigma) = true \to \mathcal{M}_{\mathcal{I}}(T)(\sigma)\downarrow \wedge \mathcal{I}(q)(\mathcal{M}_{\mathcal{I}}(T)(\sigma)) = true. \\ \hline \end{array}$$

结构化循环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

(1) B 包含一个二元谓词符号，记作 $\leq$ 。

(2) D 包含一个子集 W 且 $(W, I_0(\leq))$ 为一 WFS 。

(3) 存在 $w \in WFF_B$ 且 w 包含不多于一个变量,记为 x ,使得 $W = \{\sigma(x) | I(w)(\sigma) = true\}$ 。

循环规则为

$$\frac{(p \wedge e) \rightarrow w[x/t], [p \wedge e \wedge t = y]T_1[p \wedge t < y]}{[p]\text{while } e \text{ do } T_1 \text{ od}[p \wedge \neg e]}$$

对于赋值，组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。

§4.5 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 格式的结构化循环程序、前断言、后断言以及循环不变式作为输入，我们可以得到结构化循环程序正确的验证条件。在此基础上可以进行简化，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。

§4.6 说明

本章介绍程序与程序模型的推理验证方法。卫式迁移系统的推理验证参考 [Pel01, MP83] 。流程图程序和结构化循环程序的推理验证参考 [LS84, Fra92] 。结构化循环程序验证辅助工具 XYZ/VERI-II 的介绍参考 [Zha95] 。

参考文献

[Fra92] Nissim Francez. Program verification. Addison-Wesley Publishing Company Inc., 1992.

[LS84] Jacques Loeckx and Kurt Sieber. The foundation of program verification. John Wiley & Sons Ltd., 1984.

[MP83] 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.

[Pel01] Doron A. Peled. Software Reliability Methods. Springer-Verlag. 2001.

[Zha95] Wenhui zhang. Verification of XYZ/SE programs. Chinese Journal of Advanced Software Research 2(4):364-373, 1995.