中国科学院研究生院课程讲稿

软件系统行为与程序正确性

Software System Behavior and Program Correctness

前言

软件产品的功能对社会的各方面起着重要的作用。很多产品有着非常复杂的结构。因而软件的生产需要理论上的支持以构建可靠的软件系统。计算机科研人员在程序理论与程序正确性验证方法方面做了长期的研究，积累了丰富的程序正确性验证方面的知识。

程序正确性的体现就是给定的程序以及程序的计算环境所组成的软件系统具备我们期望的性质、不出现我们不期望出现的问题。

程序正确性验证的形式方法是基于软件系统行为模型的。为方便叙述，我们用模型代指行为模型。 通过程序与系统的建模，我们将给定的程序以及程序的计算环境所组成的软件系统抽象为模型，用形式语言进行描述，又通过形式语言的语义定义，将其描述的模型对应于软件系统的行为。 对于我们期望软件系统具备的性质，我们将这样的性质用逻辑公式表示，进而查看系统模型和这些逻辑公式的关系。 因此通过形式方法进行的程序正确性验证的主要过程为用形式语言为程序与系统建立模型、用逻辑公式描述程序性质、然后是用形式方法验证系统模型是否具备所声明的性质。

形式验证的主要问题就是对于给定的系统模型和逻辑公式，用严格的方法证明所给定的系统模型是否能够满足给定的逻辑公式。验证的方法主要包括推理验证和模型检测。 程序推理验证的理论工作在上世纪六十、七十年代取得了奠基性的成果。由于程序推理的复杂性，虽然推理验证在顺利的情况下能够证明系统满足给定的性质，但是对于不能证明的性质，推理验证在一般情况下，不能确定是性质不满足或是证明的思路不对或证明的过程过于复杂。对于并发系统而言，由于其计算过程的复杂以及系统性质的多样，基于程序推理的验证更加困难。八十年代兴起的模型检测研究试图寻求有效的算法来验证系统和系统性质的关系。模型检测已被应用于计算机硬件、软件、通信协议、控制系统、安全认证协议等领域， 成为分析、验证并发系统性质的最重要的技术。

本书结合推理验证方法和模型检测方法介绍软件系统行为模型与程序正确性验证方面的基楚知识。分五个部分：预备知识，程序与系统模型、程序逻辑、推理验证方法、模型检测方法。具体内容如下：

（ 1 ）预备知识部分包括逻辑、集合、关系、函数和有向图等内容。

（ 2 ）程序与系统模型部分包括逻辑迁移系统、显式状态迁移系统、标号迁移系统、时间迁移系统、 Petri 网、通信系统等内容。

（ 3 ）程序逻辑部分包括线性时序逻辑、分枝时序逻辑、 μ- 演算等内容。

（ 4 ）推理验证方法部分包括卫式迁移系统的推理、流程图程序的推理、结构化程序的推理等内容。

（ 5 ）模型检测方法部分包括基于状态分析的模型检测、基于路径分析的模型检测、限界模型检测等内容。

中国科学院研究生院课程讲稿

§1 预备知识

本章介绍逻辑、集合、关系和有向图方面的知识。

§1.1 命题逻辑

命题是具有确定真假意义的陈述句, 是逻辑推理的基本元素。真假值用 1 和 0 表示。 简单命题是不可分解的命题。复合命题由简单命题和联结词组成。 通常我们有一元联结词 $\neg$ (非) 和二元联结词 $\wedge$ (合取), $\vee$ (析取), $\rightarrow$ (蕴涵), $\leftrightarrow$ (等价) 等。 n- 元联结词可以看成是 $\{0,1\}^n$ 到 $\{0,1\}$ 的函数。我们有 $\neg 1 = 0$ 和 $\neg 0 = 1$ 。 用 A,B,C 等字母表示命题变元, 以下是 $\wedge$,$\vee$, $\rightarrow$,$\leftrightarrow$ 的真值表。

A	B		$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$
0	0		0	0	1	1
0	1		0	1	1	0
1	0		0	1	0	0
1	1		1	1	1	1

命题变元称为原子公式。公式集合由归纳定义生成。 设 S 是联结词的集合。由 S 生成的公式如下。 (1) 命题变元 (原子公式) 是由 S 生成的公式; (2) 若 φ 是 S 中的 0 元联结词, 则 φ 是由 S 生成的公式; (3) 若 f 是 S 中的 n 元 $(n \geq 1)$ 联结词, $\varphi_1, ..., \varphi_n$ 是由 S 生成的公式, 则 $f(\varphi_1, ..., \varphi_n)$ 是由 S 生成的公式。 这里用的是前缀记法。对于二元联结词, 我们习惯使用中缀记法。我们规定联结词的优先级以省略括号。联结词的优先级按顺序排列如下: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ 。

由全体命题变元组成的集合到 $\{0,1\}$ 的函数称为真值赋值。设 v 是真值函数。 记 A^v 为 v 赋给 A 的值。由 S 生成的公式 φ 在 v 下的值定义如下。 (1) 若 φ 是命题变元 A , 则 $v(\varphi) = A^v$; (2) 若 φ 是 S 中的 0 元联结词 c , 则 $v(\varphi) = c$; (3) 若 $\varphi = f(\varphi_1, ..., \varphi_n)$, 其中 f 是 S 中的 n 元 $(n \geq 1)$ 联结词, 则 $v(\varphi) = f(v(\varphi_1), ..., v(\varphi_n))$ 。

如果真值赋值 v 使得 $v(\varphi) = 1$, 则称 v 满足 φ , 记作 $v \models \varphi$ 。

如果有真值赋值 v , 使得 $v \models \varphi$, 则称 φ 为可满足式。否则称 φ 为永假式 (不可满足式)。如果对于每个真值赋值 v , 都有 $v \models \varphi$, 则称 φ 为永真式 (重言式)。

如果对于每个真值赋值 v , 都有 $v(\varphi) = v(\psi)$, 则称 φ 与 ψ 逻辑等价, 记作 $\varphi \Leftrightarrow \psi$ 。 $\varphi \Leftrightarrow \psi$ 当且仅当 $\varphi \leftrightarrow \psi$ 是永真式。

修改真值赋值 v 中 $A_1, ..., A_n$ 的赋值为 $a_1, ..., a_n$ 得到的赋值记作 $v[A_1/a_1, ..., A_n/a_n]$ 。设 $v' = v[A_1/a_1, ..., A_n/a_n]$ 。 我们有

$$v'(A) = \begin{cases} a_i & if \quad A = A_i, i \in \{1, ..., n\} \\ A^v & if \quad A \notin \{A_1, ..., A_n\} \end{cases}$$

用公式 $\varphi_1, ..., \varphi_n$ 分别替换公式 φ 中的不同命题变元 $A_1, ..., A_n$ 得到的公式记作 $\varphi^{\varphi_1, ..., \varphi_n}_{A_1, ..., A_n}$. 我们有

$$v(\varphi^{\varphi_1, ..., \varphi_n}_{A_1, ..., A_n}) = v[A_1/v(\varphi_1), ..., A_n/v(\varphi_n)](\varphi)$$

设 φ 是由 $\{0,1,\neg,\wedge,\vee\}$ 生成的公式。将 φ 中的 $\wedge$ 与 $\vee$ 互换、0 与 1 互换等到的公式 φ^*，称为 φ 的对偶式。 对于真值赋值 v 和其相反的真值赋值 v'，我们有 $v(\varphi)=\neg v'(\varphi^*)$。 设 φ 与 φ^* 互为对偶式， ψ 与 ψ^* 互为对偶式。如果 $\varphi\Leftrightarrow\psi$，则 $\varphi^*\Leftrightarrow\psi^*$。

逻辑公式的合取、析取和否运算满足幂等律、结合律、交换律、分配律、吸收律，德摩根律（对偶关系）。

$A\wedge A$	$=$	A	$A\vee A$	$=$	A
$A\wedge(B\wedge C)$	$=$	$(A\wedge B)\wedge C$	$A\vee(B\vee C)$	$=$	$(A\vee B)\vee C$
$A\wedge B$	$=$	$B\wedge A$	$A\vee B$	$=$	$B\vee A$
$A\wedge(B\vee C)$	$=$	$A\wedge B\vee A\wedge C$	$A\vee(B\wedge C)$	$=$	$A\vee B\wedge A\vee C$
$A\wedge(A\vee B)$	$=$	A	$A\vee(A\wedge B)$	$=$	A
$\neg(A\wedge B)$	$=$	$\neg A\vee\neg B$	$\neg(A\vee B)$	$=$	$\neg A\wedge\neg B$

设 f 是 n 元联结词， $A_1,...,A_n$ 是不同的命题变元。如果公式 φ 中不出现除 $A_1,...,A_n$ 之外的的命题变元，且 $\varphi=f(A_1,...,A_n)$，则称 φ 定义 f。 如果存在由 S 生成的公式定义 f，则称 f 可由 S 定义。

设 S 是联结词集合。若每个 n 元 $(n\geq 1)$ 联结词都可由 S 定义，则称 S 为完全集。 若 S 的任何真子集都不是完全集，则称 S 为极小完全集。 $\{\neg,\wedge,\vee\}$ 是完全集， $\{\neg,\wedge\}$ 是极小完全集。

设 Γ 为公式集合，如果真值赋值 v 满足 Γ 中的每个公式，则称 v 满足 Γ。 如果有真值赋值 v 满足 Γ，则称 Γ 是可满足的。否则称 Γ 是不可满足的。

设 Γ 为公式集合， φ 为公式。如果每个满足公式集合 Γ 的真值赋值都满足 φ，则称 φ 是 Γ 的逻辑推论，记作 $\Gamma\models\varphi$。 $\Gamma\models\varphi$ 不成立记作 $\Gamma\not\models\varphi$。 若 $\Gamma=\{\varphi_1,...,\varphi_n\}$，则将 $\Gamma\models\varphi$ 写作 $\varphi_1,...,\varphi_n\models\varphi$。

设 $\varphi_1,...,\varphi_n,\varphi,\psi$ 为公式。$\models\varphi$ 当且仅当 φ 是永真式。$\varphi_1,...,\varphi_n\models\varphi$ 当且仅当 $\varphi_1\wedge\cdots\wedge\varphi_n\rightarrow\varphi$ 是永真式。 $\varphi\Leftrightarrow\psi$ 当且仅当 $\varphi\models\psi$ 且 $\psi\models\varphi$。 $\Gamma\cup\{\varphi\}\models\psi$ 当且仅当 $\Gamma\models\varphi\rightarrow\psi$。 $\Gamma=\{\varphi_1,...,\varphi_n\}$ 是可满足的当且仅当 $\varphi_1\wedge\cdots\wedge\varphi_n$ 是可满足的。 Γ 是不可满足的当且仅当每个逻辑公式都是 Γ 的逻辑推论。

§1.2 谓词逻辑

谓词逻辑可以对所考察的命题加以细化，分清主词和谓词，考虑一般和个别情况。 谓词逻辑中使用的符号有以下几组： （1）个体变元，简称变元，有无穷多个，用 x,y,z,u,v,w 表示。 （2）个体常元，简称常元，用 a,b,c 表示。 （3）函数符号，每个符号都有与之相联系的正整数 n，并称该符号为 n 元函数符号，用 f,g,h 表示。 （4）谓词符号，每个符号都有与之相联系的正整数 n，并称该符号为 n 元谓词符号，用 A,B,C 表示。 （5）量词符号 $\forall$ 和 $\exists$。 （6）联结词符号 $\neg,\wedge,\vee,\rightarrow,\leftrightarrow$。 （7）左括号（，右括号），点。和逗号，。

设 F 是常元和函数符号的集合。由 F 生成的项定义如下。 （1）变元是由 F 生成的项； （2） F 中的常元是由 F 生成的项； （3）若 f 是 F 中的 n 元 $(n\geq 1)$ 函数符号， $t_1,...,t_n$ 是由 F 生成的项，则 $f(t_1,...,t_n)$ 是由 F 生成的项。

设 G 是谓词符号的集合。若 $t_1,...,t_n$ 是由 F 生成的项， A 是 P 中的 n 元谓词符号，则 $A(t_1,...,t_n)$ 是由 (F,G) 生成的原子公式。

$B=(F,G)$ 上的公式集合，记作 $\mathcal{L}^B$，定义如下。 （1）由 (F,G) 生成的原子公式是 $\mathcal{L}^B$ 的公式； （2）若 f 是 $\{\neg,\wedge,\vee,\rightarrow,\leftrightarrow\}$ 中的 n 元 $(n\geq 1)$ 联结词， $\varphi_1,...,\varphi_n$ 是 $\mathcal{L}^B$ 的公式，

则 $f(\varphi_1,...,\varphi_n)$ 是 $\mathcal{L}^B$ 的公式； （3）若 φ 是 $\mathcal{L}^B$ 的公式， x 是变元，则 $\forall x\varphi$ 和 $\exists x\varphi$ 是 $\mathcal{L}^B$ 的公式。

一个解释 I 由两个部分组成。其一是一个非空集合，称为论域，其二是 B 中符号到论域中的元素、函数、谓词的解释（映射）。 设 $I=(D,I_0)$ 。 对于每个常元 a ， $I_0(a)$ 为 D 中的一个元素； 对于每个 n 元函数符号 f ， $I_0(f)$ 为 D 中的一个 n 元函数； 对于每个 n 元谓词符号 P ， $I_0(P)$ 为 D 中的一个 n 元谓词。

设 I 是一个解释。从所有变元组成的集合到论域 D 的函数称为 I 中的赋值。 修改赋值 σ 中 $x_1,...,x_n$ 的赋值为 $a_1,...,a_n$ 得到的赋值记作 $\sigma[x_1/a_1,...,x_n/a_n]$ 。 我们有

$$\sigma[x_1/a_1,...,x_n/a_n](x)=\begin{cases} a_i & if \quad x=x_i, i\in\{1,...,n\} \\ \sigma(x) & if \quad x\notin\{x_1,...,x_n\} \end{cases}$$

解释和赋值共同规定了项和公式的意义。 设 σ 是 I 中的赋值。项 t 在解释 I 和赋值 σ 下的意义 $I(t)\sigma$ 定义如下。 （1）若 t 是变元 x ，则 $I(t)\sigma=\sigma(x)$ ； （2）若 t 是常元 a ，则 $I(t)\sigma=I_0(a)$ ； （3）若 t 是 $f(t_1,...,t_n)$ ，其中 f 是 n 元函数符号， $t_1,...,t_n$ 是项，则 $I(t)\sigma=I_0(f)(I(t_1)\sigma,...,I(t_n)\sigma)$ 。

设 σ 是 I 中的赋值。公式 φ 在解释 I 和赋值 σ 下的意义 $I(\varphi)\sigma$ 定义如下。 （1）若 φ 是 $P(t_1,...,t_n)$, 其中 P 是 n 元谓词符号, $t_1,...,t_n$ 是项, 则 $I(\varphi)\sigma=I_0(P)(I(t_1)\sigma,...,I(t_n)\sigma)$; （2）若 φ 是 $\neg\psi$, ψ 是公式, 则 $I(\varphi)\sigma=\neg I(\psi)\sigma$; （3）若 φ 是 $\varphi_0\wedge\varphi_1$, 则 $I(\varphi)\sigma=I(\varphi_0)\sigma\wedge I(\varphi_1)\sigma$; （4）若 φ 是 $\varphi_0\vee\varphi_1$, 则 $I(\varphi)\sigma=I(\varphi_0)\sigma\vee I(\varphi_1)\sigma$; （5）若 φ 是 $\varphi_0\to\varphi_1$, 则 $I(\varphi)\sigma=I(\varphi_0)\sigma\to I(\varphi_1)\sigma$; （6）若 φ 是 $\varphi_0\leftrightarrow\varphi_1$ ，则 $I(\varphi)\sigma=I(\varphi_0)\sigma\leftrightarrow I(\varphi_1)\sigma$; （7）若 φ 是 $\forall x\psi$ ，则 $I(\varphi)\sigma=1$ 当且仅当对于所有 $d\in D$ ， $I(\psi)\sigma[x/d]=1$ ； （8）若 φ 是 $\exists x\psi$ ，则 $I(\varphi)\sigma=1$ 当且仅当存在 $d\in D$ 使得 $I(\psi)\sigma[x/d]=1$ 。

如果公式 ψ 在公式 φ 中出现，则称 ψ 为 φ 的子公式。 变元 x 在 $\forall x\varphi$ 或 $\exists x\varphi$ 中的出现为约束出现，并称 $\forall x$ 或 $\exists x$ 的该次出现的辖域为 φ 。 如果变元 x 在 φ 中的某次出现是在 φ 的一个子公式中的约束出现，则称 x 的该次出现为在 φ 中的约束出现。 如果变元 x 在 φ 中的某次出现不是约束出现，则称该出现为在 φ 中的自由出现。 在公式 φ 中有自由出现的变元称为 φ 的自由变元，在公式 φ 中有约束出现的变元称为 φ 的约束变元。 φ 中自由变元的集合记为 $Var(\varphi)$ 。

不出现变元的项称为基项。 没有自由变元的公式称为语句。 没有约束变元的公式称为开公式。 若 $Var(\varphi)=\{x_1,...,x_n\}$ ，则称公式 $\forall x_1\cdots\forall x_n\varphi$ 为 φ 的闭包。 每个公式的闭包是一个语句，每个语句的闭包是它自己。

若 t 是基项，则对任意 σ,σ' 有 $I(t)\sigma=I(t)\sigma'$ ，即基项的意义与赋值无关。因此对于基项我们可将 $I(t)\sigma$ 简记为 $I(t)$ 。 若 φ 是语句，则对任意 σ,σ' 有 $I(\varphi)\sigma=I(\varphi)\sigma'$ 。因此对于语句我们可将 $I(\varphi)\sigma$ 简记为 $I(\varphi)$ 。

若 $x_1,...,x_n$ 是不同的变元， $t_1,...,t_n$ 是项，则称 $\{x_1/t_1,...,x_n/t_n\}$ 为代换。 若 t 是项，则 $t\{x_1/t_1,...,x_n/t_n\}$ 是用 $t_1,...,t_n$ 分别替换 t 中 $x_1,...,x_n$ 的所有出现得到的项，记为 $t^{t_1,...,t_n}_{x_1,...,x_n}$ 。若 φ 是公式，则 $\varphi\{x_1/t_1,...,x_n/t_n\}$ 是用 $t_1,...,t_n$ 分别替换 t 中 $x_1,...,x_n$ 的所有自由出现得到的公式，记为 $\varphi^{t_1,...,t_n}_{x_1,...,x_n}$ 。 如果在公式 φ 和 $\varphi^{t_1,...,t_n}_{x_1,...,x_n}$ 中变元的约束出现次数相同，则称 $t_1,...,t_n$ 对于 φ 中的 $x_1,...,x_n$ 是可代入的。 若 $t_1,...,t_n$ 对于 φ 中的 $x_1,...,x_n$ 是可代入的，则有

$$I(\varphi^{t_1,...,t_n}_{x_1,...,x_n})\sigma=I(\varphi)\sigma[x_1/I(t_1)\sigma,...,x_n/I(t_n)\sigma]$$

如果解释 I 和 I 中的赋值 σ 使得 $I(\varphi)\sigma=1$, 则称解释 I 和赋值 σ 满足 φ , 记作 $\sigma\models_I\varphi$ 。当解释给定时，简记为 $\sigma\models\varphi$ 。

如果有解释 I 和 I 中的赋值 σ 使得 $\sigma \models_I \varphi$，则称 φ 为可满足式。否则称 φ 为永假式（不可满足式）。 如果 φ 在每个解释中为真，则称 φ 为永真式（逻辑有效式）。

用谓词逻辑公式 $\varphi_1, ..., \varphi_i$ 分别替换命题逻辑公式 φ 中的命题变元 $A_1, ..., A_n$ 得到的谓词逻辑公式记为 $\varphi_{A_1,...,A_n}^{\varphi_1,...,\varphi_n}$，称为 φ 的替换实例。 命题逻辑永真式的替换实例称为重言式。

设 φ 和 ψ 是公式。如果对于每个解释 I 和 I 中的赋值 σ，$I(\varphi)\sigma = I(\psi)\sigma$，则称 φ 和 ψ 逻辑等价，记为 $\varphi \Leftrightarrow \psi$。 $\varphi \Leftrightarrow \psi$ 当且仅当 $\varphi \leftrightarrow \psi$ 是永真式。 对于 $\forall$ 和 $\exists$，我们有 $\forall x\varphi \Leftrightarrow \neg\exists x\neg\varphi$。

设 Γ 为公式集合，解释 I 和 I 中的赋值 σ 满足 Γ 中的每个公式，则称 I 和 σ 满足 Γ。 如果有解释 I 和 I 中的赋值 σ 满足 Γ，则称 Γ 是可满足的。否则称 Γ 是不可满足的。

设 Γ 为公式集合，φ 为公式。如果每个满足公式集合 Γ 的解释 I 和 I 中的赋值 σ 都满足 φ，则称 φ 是 Γ 的逻辑推论，记作 $\Gamma \models \varphi$。 $\Gamma \models \varphi$ 不成立记作 $\Gamma \not\models \varphi$。 若 $\Gamma = \{\varphi_1, ..., \varphi_n\}$，则将 $\Gamma \models \varphi$ 写作 $\varphi_1, ..., \varphi_n \models \varphi$。

设 $\varphi_1, ..., \varphi_n, \varphi, \psi$ 为公式。$\models \varphi$ 当且仅当 φ 是永真式。$\varphi_1, ..., \varphi_n \models \varphi$ 当且仅当 $\varphi_1 \wedge \cdots \wedge \varphi_n \rightarrow \varphi$ 是永真式。 $\varphi \Leftrightarrow \psi$ 当且仅当 $\varphi \models \psi$ 且 $\psi \models \varphi$。 $\Gamma \cup \{\varphi\} \models \psi$ 当且仅当 $\Gamma \models \varphi \rightarrow \psi$。 $\Gamma = \{\varphi_1, ..., \varphi_n\}$ 是可满足的当且仅当 $\varphi_1 \wedge \cdots \wedge \varphi_n$ 是可满足的。 Γ 是不可满足的当且仅当每个逻辑公式都是 Γ 的逻辑推论。

§1.3 集合

集合是由一些个体组成的整体。这些个体称为集合的元素。 集合的定义有两种：枚举定义和抽象定义。 枚举定义即是列出所有属于集合的元素。如：$A = \{a, b, c\}$。 抽象定义即是说明属于集合的元素所具有的性质特征。如：$A = \{x \in \mathbf{N} \mid x > 1\}$ 或 $x \in A \Leftrightarrow x > 1$。

两个集合相等，记作 $A = B$，当且仅当他们具有相同的元素。 集合 A 是集合 B 的子集，或说集合 A 包含于集合 B，记作 $A \subseteq B$，当且仅当所有 A 的元素都是 B 的元素。

$$\begin{array}{|ll|} \hline (A = B) & \Leftrightarrow \quad \forall x(x \in A \leftrightarrow x \in B) \\ (A \subseteq B) & \Leftrightarrow \quad \forall x(x \in A \rightarrow x \in B) \\ \hline \end{array}$$

子集关系满足以下性质。

$$\begin{array}{|l|} \hline A \subseteq A \\ A \subseteq B \text{ 且 } B \subseteq C \text{ 则 } A \subseteq C \\ A \subseteq B \text{ 且 } B \subseteq A \text{ 则 } A = B \\ \hline \end{array}$$

不含有任何元素的集合称为空集，记作 $\emptyset$。空集是最小的集合，是唯一的，它包含于任何集合之中。 由有限多个元素构成的集合称为有穷集。由无限多个元素构成的集合称为无穷集。集合 A 的全部子集的集合称为 A 的幂集，记作 $\rho(A)$。 $\rho(A) = \{X \mid X \subseteq A\}$。 若 $a \in A$，则 $\{a\} \subseteq A$。若 $A \subseteq B$，则 $A \in \rho(B)$。 有穷集合 A 的元素个数称为基数，记作 $|A|$。设 A 是有穷集合，则 $|\rho(A)| = 2^{|A|}$。

集合的运算有交、并、差。

$$\begin{array}{|ll|} \hline \text{交} & A \cap B = \{x \mid x \in A \wedge x \in B\} \\ \text{并} & A \cup B = \{x \mid x \in A \vee x \in B\} \\ \text{差} & A - B = \{x \mid x \in A \wedge x \notin B\} \\ \hline \end{array}$$

若 $A \cap B = \emptyset$，则称 A 和 B 是不相交的。 集合 A 和 B 的差集 $A - B$ 又称 B 关于 A 的相对补集。 设 U 是全集或论域，即所有与讨论相关的集合都是该全集的子集。设 A 是 U 的子集，A 关于 U 的相对补集 $U - A$，称为 A 的绝对补集，通常就称补集，记作 $\sim A$。

$A\cap A$	$=$	A	$A\cup A$	$=$	A
$A\cap(B\cap C)$	$=$	$(A\cap B)\cap C$	$A\cup(B\cup C)$	$=$	$(A\cup B)\cup C$
$A\cap B$	$=$	$B\cap A$	$A\cup B$	$=$	$B\cup A$
$A\cap(B\cup C)$	$=$	$A\cap B\cup A\cap C$	$A\cup(B\cap C)$	$=$	$A\cup B\cap A\cup C$
$A\cap(A\cup B)$	$=$	A	$A\cup(A\cap B)$	$=$	A
$\sim(A\cap B)$	$=$	$\sim A\cup\sim B$	$\sim(A\cup B)$	$=$	$\sim A\cap\sim B$

与逻辑公式的合取、析取和否运算类似，集合的交、并、补运算满足幂等律、结合律、交换律、分配律、吸收律，德摩根律。

任给两个对象 x,y，将它们按规定的顺序构成的序列，称为有序偶，记为 $\langle x,y\rangle$。有序偶有第一个元与第二个元之分，$\langle x,y\rangle$ 的第一个元是 x，第二个元是 y。有序偶可用集合表示。$\langle x,y\rangle$ 的集合表示为 $\{\{x\},\{x,y\}\}$。$\langle x,y\rangle=\langle u,v\rangle$ 当且仅当 $x=u$ 且 $y=v$。

有序偶可以推广到 n 重序偶。n 重序偶定义为 $\langle x_1,...,x_{n-1},x_n\rangle=\langle\langle x_1,...,x_{n-1}\rangle,x_n\rangle$。$\langle x_1,...,x_{n-1},x_n\rangle=\langle y_1,...,y_{n-1},y_n\rangle$ 当且仅当 $x_1=y_1$，...，$x_{n-1}=y_{n-1}$ 且 $x_n=y_n$。

集合 $A_1,...,A_n$ 的笛卡尔乘积 $A_1\times\cdots\times A_n$ 定义为 $A_1\times\cdots\times A_n=\{\langle a_1,...,a_n\rangle\mid a_1\in A_1,...,a_n\in A_n\}$。对于任意有穷集合 $A_1,...,A_n$，$|A_1\times\cdots\times A_n|=|A_1|\cdot|A_2|\cdots|A_n|$。设 $A=A_1\times\cdots\times A_n$。则第 i 分量函数 $pr_i:A\to A_i$ 定义如下。$pr_i(\langle a_1,...,a_n\rangle)=a_i$。

§1.4 关系

任何有序偶的集合称为二元关系。从 X 到 Y 的关系 R 满足 $R\subseteq X\times Y$。若 $\langle x,y\rangle\in R$，则表示成 xRy，读作 x 与 y 有关系 R。若 $\langle x,y\rangle\notin R$，则表示成 $x\bar{R}y$。一个二元关系可以用一个二元谓词确定。定义 $R=\{\langle x,y\rangle\mid P(x,y)\}$，即 xRy 当且仅当 $P(x,y)$ 成立。

设 R 是一个关系。R 中所有有序偶的第一个元的集合称为 R 的定义域，记作 $dom(R)$。R 中所有有序偶的第二个元的集合称为 R 的值域，记作 $ran(R)$。集合 X 到 X 的关系称为 X 上的二元关系。关系的性质由关系中包含的所有有序偶所确定。记 $\forall x_1\cdots\forall x_n(x_1\in X\wedge\cdots\wedge x_n\in X\to\varphi)$ 为 $\forall x_1,...,x_n\in X.\varphi$。设 R 是非空集合 X 上的关系。

自反性	$\forall x\in X.(xRx)$
反自反性	$\forall x\in X.(x\bar{R}x)$
对称性	$\forall x,y\in X.(xRy\to yRx)$
反对称性	$\forall x,y\in X.(xRy\wedge yRx\to x=y)$
传递性	$\forall x,y,z\in X.(xRy\wedge yRz\to xRz)$

如果 R 和 S 是 X 到 Y 的二元关系，则 $R\cap S$，$R\cup S$，$R-S$，$\sim S$ 都是 X 到 Y 的二元关系，且

$x(R\cap S)y$	$\Leftrightarrow$	$xRy\wedge xSy$
$x(R\cup S)y$	$\Leftrightarrow$	$xRy\vee xSy$
$x(R-S)y$	$\Leftrightarrow$	$xRy\wedge x\bar{S}y$
$x(\sim S)y$	$\Leftrightarrow$	$x\bar{S}y$

设 R 是 X 到 Y 的关系。R 的逆关系是 Y 到 X 的关系，记作 R^{-1}，定义为 $R^{-1}=\{\langle y,x\rangle\in Y\times X\mid\langle x,y\rangle\in R\}$。

设 R 是 X 到 Y 的关系，S 是 Y 到 Z 的关系。$R\circ S$ 是 X 到 Z 的关系，称为 R 和 S 的复合关系，定义为 $R\circ S=\{\langle x,z\rangle\in X\times Z\mid\exists y\in Y.(xRy\wedge ySz)\}$。$\circ$ 称为关系的复合运算。在

不引起混淆的情况下，复合关系的运算符常省略不写。关系的复合运算满足结合律。 设 R 是 X 到 Y 的关系， S 是 Y 到 Z 的关系。则有 $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$ 。

设 R 是 X 上的二元关系， $n \in \mathbf{N}$ 。 R 的 n 次幂，记作 R^n ，定义如下。 $R^0 = I_X = \{\langle x, x\rangle \mid x \in X\}$ 是集合 X 上的恒等关系； $R^{n+1} = R^n \circ R$ 。

设 R 是 X 上的二元关系，关系 R' 是 R 的自反（对称、传递）闭包当且仅当 （1） R' 是自反（对称、传递）的； （2） $R \subseteq R'$ ； （3）对于任何自反（对称、传递）关系 R'' ，如果 $R \subseteq R''$ ，则 $R' \subseteq R''$ 。 用 $r(R), s(R), t(R)$ 分别表示 R 的自反闭包、对称闭包、传递闭包。我们有

$r(R)$	$=$	$R \cup I_X$
$s(R)$	$=$	$R \cup R^{-1}$
$t(R)$	$=$	$\bigcup_{i=1}^{\infty} R^i$

为书写方便，关系 R 的传递闭包通常记为 R^+ 。 R 的自反传递闭包通常记为 R^* 。

满足自反性、对称性和传递性的一个非空集合上的关系称为等价关系。 设 R 是集合 X 上的等价关系。对于任意 $x \in X$ ，集合 $[x]_R$ 定义如下： $[x]_R = \{y \in X \mid yRx\}$ 。 称 $[x]_R$ 为由 x 所代表的等价类。用 X/R 表示 R 等价类的集合 $\{[x]_R \mid x \in X\}$ ，称 X/R 为 X 模 R 的商集。

设 A 是非空集合 S 子集的聚合。对于每个集合 $B \in A$ ， $B \neq \emptyset$ 且 $\bigcup A = S$ 。则称集合 A 是 S 的一个覆盖。 若 A 是 S 的一个覆盖，且对任意 $B, C \in A$ ， $B \neq C$ 则 $B \cap C = \emptyset$ ，则称 A 是 S 的一个划分。

任何一个 X 上的等价关系 R 都定义了 X 的一个划分，即 X/R 。 任何 X 的一个划分 $A = \{A_1, ..., A_n\}$ 都定义了一个等价关系 R ，即 xRy 当且仅当 x, y 同在一个 A_i 中。

满足自反性、反对称性和传递性的一个非空集合上的关系称为偏序关系。 如果 $\leq$ 是 X 上的偏序关系，那么有序偶 $\langle P, \leq\rangle$ 表示偏序集合。

设 $\langle P, \leq\rangle$ 是偏序集合。如果对于每一个 $x, y \in P$ 都有 $x \leq y$ 或 $y \leq x$ ，则称 $\leq$ 上为 P 上的全序或线序，称 $\langle P, \leq\rangle$ 为全序集合或链。

设 $\langle P, \leq\rangle$ 是偏序集合，并且 $A \subseteq P$ 。设 $a \in A, b \in P$ 。

a 为 A 的最大元：	$\forall x \in A.(x \leq a)$
a 为 A 的最小元：	$\forall x \in A.(a \leq x)$
a 为 A 的极大元：	$\forall x \in A.(a \neq x \rightarrow a \not\leq x)$
a 为 A 的极小元：	$\forall x \in A.(a \neq x \rightarrow x \not\leq a)$
b 为 A 的上界：	$\forall x \in A.(x \leq b)$
b 为 A 的下界：	$\forall x \in A.(b \leq x)$
b 为 A 的最小上界：	b 是 A 的上界，且对于每一个 A 的上界 b' ，有 $b \leq b'$
b 为 A 的最大下界：	b 是 A 的下界，且对于每一个 A 的下界 b' ，有 $b' \leq b$

一个偏序集合的最大元，最小元，最小上界，最大下界，如果存在，则是唯一的。 最小上界，也称上确界，记作 $\sqcup$ ， lub 或 sup ，最大下界，也称下确界，记作 $\sqcap$ ， glb 或 inf 。

一个偏序集合 $\langle P, \leq\rangle$ ，如果它的每一个非空子集都有一个最小元，则称 $\leq$ 为良序的，称 $\langle P, \leq\rangle$ 为良序集合。 每一个良序集合都是全序集合，但全序集合未必都是良序集合，而每一个有限的全序集合都是良序集合。

一个偏序集合 $\langle P, \leq\rangle$ ，如果它的每一个非空子集都有一个极小元，则称 $\leq$ 为良基的 (Well Founded) ，称 $\langle P, \leq\rangle$ 为良基集合。 一个集合是良基集合当且仅当该集合中不存在无限递减的序列。

设 $\langle P,\leq\rangle$ 为良基集合。记 $x\leq y$ 且 $x\neq y$ 为 $x<y$ 。以下推理规则为良基集合上的归纳法 (Noetherian Induction) 。

$$若\ \forall x'\in P.(\forall x\in P.(x<x'\to\varphi(x))\to\varphi(x'))\ 则\ \forall x\in P.\varphi(x)\ 。$$

设 $\langle P_1,\leq_1\rangle,...,\langle P_n,\leq_n\rangle$ 为偏序，$P=P_1\times\cdots\times P_n$ 。则偏序 $\langle P_1,\leq_1\rangle,...,\langle P_n,\leq_n\rangle$ 的笛卡尔积 $\langle P,\leq\rangle$ 是一个偏序，其中 $\leq$ 定义如下。 对所有 $a,b\in P$ ， $a\leq b$ 当且仅当对所有 $i\in\{1,...,n\}$ ，有 $pr_i(a)\leq_i pr_i(b)$ 。 设 $S\subseteq P_1\times\cdots\times P_n$ 。则 $\sqcup S$ 存在当且仅当对于所有 $i\in\{1,...,n\}$ ，$\sqcup pr_i(S)$ 存在，且 $\sqcup S=\langle\sqcup pr_1(S),...,\sqcup pr_n(S)\rangle$ 。

§1.5 函数

设 f 是集合 X 到集合 Y 的关系。如果对每一个 $x\in X$ 存在唯一的 $y\in Y$ 使得 $\langle x,y\rangle\in f$ ，则称 f 为 X 到 Y 的一个函数。记为 $f:X\to Y$ 。 X 称为 f 的定义域， Y 称为 f 的值域。

对于函数 $f:X\to Y$ ，如果 $\langle x,y\rangle\in f$ ，则称 x 为自变量， y 是函数 f 在 x 处的值，也称 y 为在 f 作用下 x 的象，而称 x 为 y 的一个象源。通常用 $y=f(x)$ 表示 $\langle x,y\rangle\in f$ 。

设函数 $f:X\to Y$ 且 $A\subseteq X$ ，则 $f\cap(A\times Y)$ 是从 A 到 Y 的函数，称为 f 受限制于 A ，记为 $f|A$ 。集合 $\{f(x)\mid x\in A\}$ 称为 A 在 f 下的象，记为 $f(A)$ 。

若 $X'\subseteq X$ ，且 $f:X'\to Y$ 是 X' 到 Y 的函数，则称 f 为 X 到 Y 的偏函数。为区别于偏函数，函数又称全函数。

设 $f:X\to Y$ 和 $g:Y\to Z$ 是两个函数，则 f 和 g 的复合函数 $g\circ f$ 是一个从 X 到 Z 的函数，且对于所有的 $x\in X$ ， $(g\circ f)(x)=g(f(x))$ 。 函数的复合满足结合律。

若 $f:X\to X$ 是一个函数，则 f 能够对自身复合任意多次。 f 对自身复合任意多次的定义如下。 $f^0(x)=I_X(x)$ ； $f^{n+1}(x)=f(f^n(x))$ 。

记 $\exists x_1\cdots\exists x_n(x_1\in X\wedge\cdots\wedge x_n\in X\wedge\varphi)$ 为 $\exists x_1,...,x_n\in X.\varphi$ 。设 $f:X\to Y$ 是一个函数。

f 是满射的	$\forall y\in Y.\exists x\in X.(f(x)=y)$
f 是入射的	$\forall x_1,x_2\in X.(f(x_1)=f(x_2)\to x_1=x_2)$
f 是双射的	f 是满射的且是入射的

若 $f:X\to Y$,$g:Y\to Z$ 都是满射函数，则 $g\circ f$ 也是满射函数； 若 $f:X\to Y$,$g:Y\to Z$ 都是入射函数，则 $g\circ f$ 也是入射函数； 若 $f:X\to Y$,$g:Y\to Z$ 都是双射函数，则 $g\circ f$ 也是双射函数。

设 f 是双射的。它的反函数是 f 的逆关系，记作 f^{-1} 。 若 $f:X\to Y$ 是双射的，则其反函数 $f^{-1}:Y\to X$ 也是双射的。 若 $f:X\to Y$,$g:Y\to Z$ 都是双射函数，则 $(g\circ f)^{-1}$ 也是双射函数，且 $(g\circ f)^{-1}=f^{-1}\circ g^{-1}$ 。

设 U 是全集， $A\subseteq U$ 。 A 的特征函数 $\Psi_A:U\to\{0,1\}$ 定义如下。

$$\Psi_A(x)=\begin{cases}1 & if\quad x\in A\\ 0 & if\quad x\notin A\end{cases}$$

设 U 是全集， $A,B\subseteq U$ 。则对所有 $x\in U$ ，以下等式成立。

$\Psi_{A\cap B}(x)$	$=$	$\Psi_A(x)\cdot\Psi_B(x)$
$\Psi_{A\cup B}(x)$	$=$	$\Psi_A(x)+\Psi_B(x)-\Psi_{A\cap B}(x)$
$\Psi_{\sim A}(x)$	$=$	$1-\Psi_A(x)$

X 到 Y 的所有全函数的集合记作 $(X\to Y)$ 。 设 X 是一个集合， $\langle Y,\leq\rangle$ 是一个偏序，$f,g:X\to Y$ 是两个函数。 $f\leq g$ 当且仅当对于所有 $x\in X$ ， $f(x)\leq g(x)$ 。 $\langle(X\to Y),\leq\rangle$ 构成一个偏序。

设 $S \subseteq (X \to Y)$ 是一个 X 到 Y 的函数的集合。设 $x \in X$ 。 Y 的子集 $\{f(x) \mid f \in S\}$ 记作 $S(x)$ 。$\sqcup S$ 存在当且仅当对于所有 $x \in X$ ，$\sqcup S(x)$ 存在，且对于所有 $x \in X$ ，$(\sqcup S)(x) = \sqcup S(x)$ 。

§1.6 完全偏序和格上的不动点

一个偏序 $\langle X, \leq \rangle$ ，如果它 X 有最小元，且对于 X 上的每一条链 $S \subseteq X$ ， $\sqcup S$ 都存在，则称 $\langle X, \leq \rangle$ 为全偏序。 X 的最小元通常记作 $\bot_X$ 或 $\bot$ 。

任何有最小元且只包含有穷链的偏序是完全偏序。如果 $\langle P_1, \leq_1 \rangle, ..., \langle P_n, \leq_n \rangle$ 是完全偏序，则其笛卡尔积 $\langle P_1 \times \cdots \times P_n, \leq \rangle$ 是完全偏序。 如果 X 是一个集合， $\langle Y, \leq \rangle$ 是一个完全偏序，则 $\langle (X \to Y), \leq \rangle$ 是完全偏序。

设 $\langle X, \leq \rangle$ 是一个完全偏序， $Y \subseteq X$ 。 $\langle Y, \leq \rangle$ 是 $\langle X, \leq \rangle$ 的子完全偏序，当且仅当 $\langle Y, \leq \rangle$ 是一个完全偏序，且对于 Y 上的每一条链 S ，都有 $\sqcup_Y S = \sqcup_X S$ 。

设 $\langle X, \leq_1 \rangle$ 和 $\langle Y, \leq_2 \rangle$ 是偏序， $f : X \to Y$ 是函数。f 是单调的当且仅当 $\forall x_1, x_2 \in X.(x_1 \leq_1 x_2 \to f(x_1) \leq_2 f(x_2))$ 。

设 $\langle X, \leq_1 \rangle$ 是偏序， $\langle Y, \leq_2 \rangle$ 是完全偏序。从 X 到 Y 的单调函数的集合构成 $\langle (X \to Y), \leq_2 \rangle$ 的一个子完全偏序。

设 $\langle X, \leq_1 \rangle$ 和 $\langle Y, \leq_2 \rangle$ 是完全偏序， $f : X \to Y$ 是函数。 f 是连续的当且仅当对 X 的每一条链 $S \subseteq X$ ， $\sqcup f(S)$ 都存在，且 $\sqcup f(S) = f(\sqcup S)$ 。 连续函数的集合记作 $[X \to Y]$ 。

设 $\langle X, \leq_1 \rangle$ 和 $\langle Y, \leq_2 \rangle$ 是完全偏序。从 X 到 Y 的连续函数的集合 $[X \to Y]$ 构成 $\langle (X \to Y), \leq \rangle$ 的一个子完全偏序。

设 $\langle X, \leq_1 \rangle$ 和 $\langle Y, \leq_2 \rangle$ 是完全偏序， $f : X \to Y$ 是函数。 f 是连续的当且仅当 f 是单调的且对 X 的每一条链 $S \subseteq X$ ， $f(\sqcup S) \leq \sqcup f(S)$ 。 如果 X 只包含有穷链，则 f 是连续的当且仅当 f 是单调的。

设 $\langle X, \leq \rangle$ 是偏序， $f : X \to X$ 是函数。若 $f(x) = x$ ，则称 x 是 f 的不动点。 若 x 是 f 的不动点，且对任意 f 的不动点 y ，都有 $x \leq y$ ，则称 x 是 f 的最小不动点。 f 的最小不动点记作 μf 。 若 x 是 f 的不动点，且对任意 f 的不动点 y ，都有 $y \leq x$ ，则称 x 是 f 的最大不动点。 f 的最大不动点记作 νf 。

为了能够清楚地说明 f 的受不动点标记约束的自变量是 x ， μf 有时写成 $\mu x.f$ ， νf 写成 $\nu x.f$ 。

设 $\langle X, \leq \rangle$ 是完全偏序， $f : X \to X$ 是连续函数。则 f 有最小不动点，且

$$\mu f = \sqcup\{f^i(\bot) \mid i \in \mathbf{N}\}$$

设 $\langle X, \leq \rangle$ 是完全偏序， $f : X \to X$ 是连续函数， $x \in X$ 。若 $f(x) \leq x$ ，则 $\mu f \leq x$ 。

设 $\langle X, \leq \rangle$ 是完全偏序， $\varphi : X \to \{0, 1\}$ 是一个谓词。 φ 是相容的当且仅当对 X 的每一条链 S ，若 $\bigwedge_{x \in S} \varphi(x)$ 为真，则 $\varphi(\sqcup S)$ 为真。

设 $\langle X, \leq \rangle$ 是完全偏序， $\varphi : X \to \{0, 1\}$ 是一个相容谓词。以下推理规则为不动点归纳法。

$$\text{若 } \varphi(\bot) \text{ 且 } \forall x \in X.(\varphi(x) \to \varphi(f(x))) \text{ ，则 } \varphi(\mu f) \text{ 。}$$

一个偏序 $\langle X, \leq \rangle$ ，如果任意两个 X 中的元素都有最小上界和最大下界，则称 $\langle X, \leq \rangle$ 为格。

一个偏序 $\langle X, \leq \rangle$ ，如果任 X 中的任意子集都有最小上界，则称 $\langle X, \leq \rangle$ 为完全格。一个偏序 $\langle X, \leq \rangle$ 是完全格当且仅当其任意子集都有最大下界。

设 $\langle X, \leq \rangle$ 是完全格， $f : X \to X$ 是单调函数。则 f 有非空的不动点集，且该集构成一个完全格， f 有最小和最大不动点。

$$\mu f = \sqcap\{x \in X \mid f(x) \leq x\}$$
$$\nu f = \sqcup\{x \in X \mid x \leq f(x)\}$$

§1.7 有向图

有向图 D 是一有序偶 $\langle V,E\rangle$，其中 V 是一非空集合，E 是 V 上的一个二元关系。分别称 V 和 E 为有向图 D 的顶点的集合和边的集合。为表示的直观性，边的集合 E 有时写成 $\rightarrow$。

设有两个图 $D=\langle V,E\rangle$ 和 $D'=\langle V',E'\rangle$。若 $V\subseteq V'$ 且 $E\subset E'$，则称 D 为 D' 的子图，并表示为 $D\subseteq D'$。 若 $V\subseteq V'$ 且 $E=\{\langle u,v\rangle\in E'\mid u,v\in V\}$，则称 D 为 D' 的导出子图。

在有向图 $D=\langle V,E\rangle$ 中，把边的一个序列 $(e_1,...,e_n)$ 称为一条通路，其中 $e_i=\langle v_i,v_{i+1}\rangle\in E$（$i=1,...,n$）。 通路 $(e_1,...,e_n)$ 的长度为出现在通路中的边的次数，记作 $|(e_1,...,e_n)|$。通路 $(e_1,...,e_n)$ 通常也用顶点序列 $(v_1,...,v_{n+1})$ 表示。

对于有向图的一条通路，如果每条边的出现不超过一次，则称该通路为简单通路。如果每个顶点的出现不超过一次，则称该通路为基本通路。基本通路一定是简单通路。通过有向图中所有顶点的通路称为完备通路。

如果一条通路的开始和终结于同一顶点，则称该通路为回路。如果该回路中每条边的出现不超过一次，则称该回路为简单回路。如果该回路除去最后一个点的通路中每个顶点的出现不超过一次，则称该回路为基本回路。通过有向图中所有顶点的回路称为完备回路。

如果存在从顶点 u 到顶点 v 的通路，则称顶点 u 可以到达顶点 v，即 u,v 满足 $u\rightarrow^+ v$。如果存在从顶点 u 到顶点 v 的通路，则存在从顶点 u 到顶点 v 的基本通路。 在一个有 n 个顶点的有向图中，任何基本通路的长度都不超过 $n-1$，任何基本回路的长度都不超过 n。

一个有向图 $D=\langle V,E\rangle$，如果对它的每一对顶点 u 和 v，可以从 u 到达 v 并且可以从 v 到达 u，则称 D 为强连通图。有向图 D 是强连通的当且仅当 D 有完备回路。

在有向图 D 中，$A\subseteq D$ 是 D 的一个极大强连通导出子图，当且仅当 A 是 D 的导出子图，A 是强连通的，且对于任意的 D 的强连通的子图 $B\subseteq D$，若 $A\neq B$ 则 $A\not\subseteq B$。 在有向图 D 中，D 的一个极大强连通导出子图，称为 D 的一个强连通分图。

设 $D=\langle V,E\rangle$ 为有向图。设 $A\subseteq V$。从集合 A 可达的顶点的集合（包括 A）为 $\{b\mid a\rightarrow^* b,a\in A\}$，记为 $rh(A)$。可达 A 的顶点的集合为 $\{b\mid b\rightarrow^* a,a\in A\}$，记为 $rh^b(A)$。

若 D 是强连通图，则 $rh(A)=rh^b(A)=V$。

定义 $E(a)=\{b\mid E(a,b)\},E(A)=\{b\mid E(a),a\in A\}$。设 V 是有穷集合，$A\subseteq V$。将 E 和 E^{-1} 看成是 $(\rho(V)\rightarrow\rho(V))$ 中的函数，将 $A\cup E(Z)$ 和 $A\cup E^{-1}(Z)$ 看成是自变量为 Z 的 $(\rho(V)\rightarrow\rho(V))$ 中的函数，则

$$\begin{array}{ll} rh(A) & =\bigcup_{i=0}^{\infty}E^i(A) \qquad =\mu Z.(A\cup E(Z)).\\ rh^b(A) & =\bigcup_{i=0}^{\infty}(E^{-1})^i(A) \quad =\mu Z.(A\cup E^{-1}(Z)).\end{array}$$

设 $D_1=\langle V_1,E_1\rangle$ 和 $D_2=\langle V_2,E_2\rangle$ 为有向图。

一个关系 $\sigma\subseteq V_1\times V_2$ 称为 D_1 到 D_2 的模拟关系，如果对任意 $(u,v)\in\sigma$，对任意 $u'\in V_1$，如果 $E_1(u,u')$，则存在 $v'\in V_2$，$E_2(v,v')$ 且 $(u',v')\in\sigma$。σ 是模拟关系当且仅当 $\sigma^{-1}E_1\subseteq E_2\sigma^{-1}$。

一个关系 $\sigma\subseteq V_1\times V_2$ 称为 D_1 和 D_2 上的互模拟关系，如果对任意 $(u,v)\in\sigma$，（1）对任意 $u'\in V_1$，如果 $E_1(u,u')$，则存在 $v'\in V_2$，$E_2(v,v')$ 且 $(u',v')\in\sigma$；（2）对任意 $v'\in V_2$，如果 $E_2(v,v')$，则存在 $u'\in V_1$，$E_1(u,u')$ 且 $(u',v')\in\sigma$。 σ 是互模拟关系当且仅当 $\sigma^{-1}E_1\subseteq E_2\sigma^{-1}$ 且 $\sigma E_2\subseteq E_1\sigma$。

设 $D=\langle V,E\rangle$ 为有向图。

$\sigma\subseteq V\times V$ 称为 D 上的模拟等价关系，如果对任意 $(u,v)\in\sigma$，存在一个模拟关系 $\sigma'\subseteq V\times V$ 使得 $(u,v)\in\sigma'$ 且存在一个模拟关系 $\sigma''\subseteq V\times V$ 使得 $(v,u)\in\sigma''$。 $\sigma\subseteq V\times V$ 称为 D 上的互模拟等价关系，如果 σ 是 V 上的最大的互模拟关系。

§1.8 说明

本章的目的是让读者对离散数学中与程序验证相关的基础概念有所了解。离散数学的相关内容参考 [Lu92, YHXT98, ZLL03] 。

参考文献

[Lu92] 陆汝钤。计算机语言的形式语义。科学出版社。 1992 。

[YHXT98] 尹宝林、何自强、许光汉、檀风琴。离散数学。高等教育出版社。 1998 。

[ZLL03] 左孝凌、李为镒、刘永才。离散数学。上海科学技术文献出版社。 2003 。

中国科学院研究生院课程讲稿

§2 程序与系统模型

程序的正确从广义上说就是包含该程序的系统的运行过程满足给定的性质。要说明程序的正确，我们需要描述系统的这种运行过程，即系统的运行模型，简称为系统模型。本章介绍系统模型。

§2.1 逻辑迁移模型

隐式迁移关系的描述通常基于一阶逻辑。 给定常元和函数符号集合 F ，谓词符号集合 P 。记 $B=(F,P)$ 上一阶逻辑公式集合为 WFF_B ， B 上的项的集合为 T_B 。记 WFF_B 中不带量词的公式集合为 QFF_B 。 系统的隐式迁移模型通常是在这样的一阶逻辑上增加符号进行赋值操作和流程控制，其中 WFF_B 中的公式可作为流程控制的条件， T_B 中的项可作为赋值语句中值的表示。

§2.1.1 结构化循环语句模型

结构化循环语句模型是一阶逻辑的扩充。 重要模型要素有赋值、顺序复合、条件语句和循环语句。 给定 $B=(F,P)$ 。我们增加以下集合的符号：

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

Definition 2.1 给定 $B=(F,P)$ 和变量集合 V 。一个 (B,V) 上的结构化循环语句模型是一个字符串，其集合 S 定义如下：

S	$::=$	$\epsilon \mid T;\epsilon$
T	$::=$	$x:=t \mid$
		$T;T \mid$
		$if\ e\ then\ T\ else\ T\ fi \mid$
		$while\ e\ do\ T\ od$

其中 $x\in V$ 为变元， $t\in T_B$ 为 B 上的项且 $Var(t)\subseteq V$ ， $e\in QFF_B$ 为一个不带量词的公式且 $Var(e)\subseteq V$ 。

给定 $B=(F,P)$ 和变量集合 V 。 (B,V) 上的结构化循环语句模型的集合记为 $\mathcal{L}_{\bigcirc}^{(B,V)}$ 。为书写方便，在不引起歧义的情况下，模型结尾的 ϵ 可以省略不写。

系统状态

模型的系统状态有两个部分，即模型运行的剩余部分和变量状态。 变量状态空间为 V 中变量取值的组合 $\Sigma=\{\sigma|\sigma(x)\in D,x\in V\}$ 。 系统的状态空间为模型和 V 中变量取值的组合 $\mathcal{L}_{\bigcirc}^{(B,V)}\times\Sigma$ 。

状态迁移关系

一个模型的运行取决于 B 的解释。 给定 B 的一个解释 I 。模型系统状态的迁移关系 $\rightarrow$ 为 $(\mathcal{L}_{\bigcirc}^{(B,V)}\times\Sigma)^2$ 的一个子集。 $(S_0,\sigma_0)\rightarrow(S_1,\sigma_1)$ 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; 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; S'	$\sigma_0 \models_I e$	S; while (e) do S od; S'	σ_0
while e do S od; S'	$\sigma_0 \not\models_I e$	S'	σ_0
ϵ		ϵ	σ_0

以上的最后一项说明模型运行终止后状态保持不变。

运行

用 $[(S_i, \sigma_i)]_{i\geq 0}$ 表示无穷系列

$$(S_0, \sigma_0)(S_1, \sigma_)(S_2, \sigma_2).....$$

模型 T 的一个运行是状态集 $\mathcal{L}_{\bigcirc}^{(B,V)} \times \Sigma$ 上的一个无穷序列 $[(S_i, \sigma_i)]_{i\geq 0}$ 满足 $S_0 = T$ 且对任意 $i \geq 0$,

$$(S_i, \sigma_i) \rightarrow (S_{i+1}, \sigma_{i+1})$$

模型 T 在初始变量状态为 σ_0 时运行可到达的状态的集合为 $\{(T', \sigma) | (T, \sigma_0) \xrightarrow{*} (T', \sigma)\}$ 。

终止

模型 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 $(T, \sigma) \xrightarrow{*} (\epsilon, \sigma')$ 。

模型性质

模型的基本性质包括模型运行的终止性以及模型的终止状态和初始状态的关系。设 φ 和 ψ 是谓词。一些基本的性质描述如下：

（ 1 ）给定初始时的变量状态满足的条件 φ ，给定 ψ 。要求如果终止，则模型的终止时的变量状态满足 ψ 。这个要求称为模型对于 φ 和 ψ 的部分正确性。 φ 称为模型的前断言， ψ 称为模型的后断言。

（ 2 ）给定初始时的变量状态满足的条件 φ ，给定 ψ 。要求模型能够终止并且模型的终止时的变量状态满足 ψ 。这个要求称为模型对于 φ 和 ψ 的完全正确性。

（ 3 ）完全正确性包括了模型的终止性，即给定初始时的变量状态满足的条件 φ ，要求模型能够终止。

模型性质的形式定义

设谓词 φ 为模型 T 的初始状态满足的条件。 T 对于谓词 φ 的终止性定义如下：

$$\text{终止性：} \quad \varphi(\sigma) \rightarrow ((T, \sigma) \xrightarrow{*} (\epsilon, \sigma'))$$

设谓词 φ 为模型 T 的初始状态满足的条件，谓词 ψ 为对模型 T 的终止状态的要求。 T 对于谓词 φ 和谓词 ψ 的部分正确性和完全正确性定义如下：

$$\text{部分正确性：} \quad \varphi(\sigma) \rightarrow ((T, \sigma) \xrightarrow{*} (\epsilon, \sigma') \rightarrow \psi(\sigma'))$$
$$\text{完全正确性：} \quad \varphi(\sigma) \rightarrow ((T, \sigma) \xrightarrow{*} (\epsilon, \sigma') \wedge \psi(\sigma'))$$

相应地可以定义以公式为前后断言的模型正确性。 对于公式 φ 和 ψ ，可以把公式通过 I 解释成为谓词，则可以有类似的性质描述。 T 对于公式 φ 的终止性即 T 对于谓词 $I(\varphi)$ 的终止性即

$$\text{终止性：}\quad I(\varphi)(\sigma) \to ((T,\sigma)\xrightarrow{*}(\epsilon,\sigma'))$$

T 对于公式 φ 和公式 ψ 的部分正确性和完全正确性定义如下：

$$\begin{array}{ll}\text{部分正确性：} & I(\varphi)(\sigma) \to ((T,\sigma)\xrightarrow{*}(\epsilon,\sigma') \to I(\psi)(\sigma')) \\ \text{完全正确性：} & I(\varphi)(\sigma) \to ((T,\sigma)\xrightarrow{*}(\epsilon,\sigma') \wedge I(\psi)(\sigma'))\end{array}$$

T 对于公式 φ 和公式 ψ 的部分正确性和完全正确性分别记作 $\models_I \{\varphi\}T\{\psi\}$ 和 $\models_I [\varphi]T[\psi]$ 。

§2.1.2 流程图模型

流程图模型 是一阶逻辑的扩充。 流程图模型主要有两种语句。一种是赋值语句。一种是条件语句。这些语句称为指令。 在一阶逻辑的基础上，我们增加以下两个集合的符号：

$$\begin{array}{ll}\text{辅助符号集 AX} & \{:=, :, goto, if, else\} \\ \text{标号符号集 LB} & \{beg, end, l_1, l_2, ...,\}\end{array}$$

给定 $B=(F,P)$ 和变量集合 V 。定义 (B,V) 上的流程图模型的两种指令如下：

l_1: $(x_1,...,x_n) := (t_1,...,t_n)$ goto l_2
l_1: if (e) goto l_2 else goto l_3

其中 $l_1,l_2,l_3 \in LB$ 为标号， $l_1 \neq end$, $t_1,...,t_n \in T_B$ 且 $\bigcup_{i=1}^{n} Var(t_i) \subseteq V$, $e \in QFF_B$ 且 $Var(e) \subseteq V$, $x_1,...,x_n \in V$ 且对 $1 \leq i < j \leq n$, $x_i \neq x_j$ 。 冒号左边的标号称为被定义标号。冒号右边的句子称为标号定义。

Definition 2.2 一个 (B,V) 上的结流程图模型 T 为满足以下条件的指令集合。

标号 beg 必须有定义
标号 end 不被定义
出现在标号定义中的除 end 外的标号必须有定义
每个标号最多被定义一次

给定 $B=(F,P)$ 和变量集合 V 。(B,V) 上流程图模型的集合记为 $\mathcal{L}_{\hookleftarrow}^{(B,V)}$ 。记 $\mathcal{L}_{\hookleftarrow}^{B}$ 为任给变量集合 V 的 $\mathcal{L}_{\hookleftarrow}^{(B,V)}$ 的并集。

系统状态

流程图模型的系统状态有两个部分：即标号和变量状态。标号可以理解为模型或系统的控制状态。 变量状态空间为 V 中变量取值的组合 $\Sigma = \{\sigma | \sigma(x) \in D, x \in V\}$ 。 系统的状态空间为标号和 V 中变量取值的组合 $LB \times \Sigma$ 。

状态迁移关系

给定 B 的一个解释 I 。给定指令 t 。 $(l_i,\sigma_i)\xrightarrow{t}(l_{i+1},\sigma_{i+1})$ 当且仅当以下一种情况成立。

t 为 l_i: $(v_1,...,v_n) := (t_1,...,t_n)$ goto l_{i+1} ，
且 $\sigma_{i+1} = \sigma_i[v_1/I(t_1)(\sigma_i)]\cdots[v_n/I(t_n)(\sigma_i)]$
t 为 l_i: (if (e) goto l' else goto l'' ，
$\sigma_{i+1} = \sigma_i$ 且 $\sigma_i \models_I e$ 则 $l_{i+1} = l'$ ， 否则 $l_{i+1} = l''$

给定流程图模型 T 和 B 的一个解释 I 。我们定义流程图模型系统状态的转换关系 $\xrightarrow{T}$ 如下。 $(l_i,\sigma_i)\xrightarrow{T}(l_{i+1},\sigma_{i+1})$ 当且仅当以下一种情况成立。

存在指令 $t \in T$ 使得 $(l_i, \sigma_i) \xrightarrow{t} (l_{i+1}, \sigma_{i+1})$ 或
不存在定义标号 l_i 的指令且 $(l_i, \sigma_i) = (l_{i+1}, \sigma_{i+1})$ 。

在不引起混淆的时候，我们将 $\xrightarrow{T}$ 中的 T 省略写成 $\rightarrow$ 。

运行

用 $[(l_i, \sigma_i)]_{i \geq 0}$ 表示无穷系列

$$(l_0, \sigma_0)(l_1, \sigma_1)(l_2, \sigma_2).....$$

流程图模型 T 的一个运行是状态集 $LB \times \Sigma$ 上的一个无穷序列 $[(l_i, \sigma_i)]_{i \geq 0}$ 满足 $l_0 = beg$ 且对任意 $i \geq 0$, $(l_i, \sigma_i) \rightarrow (l_{i+1}, \sigma_{i+1})$ 。

流程图模型 T 在初始变量状态为 σ_0 时运行可到达的状态的集合为 $\{(l, \sigma) | (beg, \sigma_0) \xrightarrow{*} (l, \sigma)\}$ 。

终止

模型在初始变量状态为 σ 时的运行终止，当且仅当存在 σ' 使得 $(beg, \sigma) \xrightarrow{*} (end, \sigma')$ 。

模型性质

设谓词 φ 为模型 T 的初始状态满足的条件。 T 对于 φ 的终止性定义如下：

$$\text{终止性：} \quad \varphi(\sigma) \rightarrow ((beg, \sigma) \xrightarrow{*} (end, \sigma'))$$

设谓词 φ 为模型 T 的初始状态满足的条件，谓词 ψ 为对模型 T 的终止状态的要求。 T 对于 φ 和 ψ 的部分正确性和完全正确性定义如下：

$$\begin{aligned} &\text{部分正确性：} \quad \varphi(\sigma) \rightarrow ((beg, \sigma) \xrightarrow{*} (end, \sigma') \rightarrow \psi(\sigma')) \\ &\text{完全正确性：} \quad \varphi(\sigma) \rightarrow ((beg, \sigma) \xrightarrow{*} (end, \sigma') \wedge \psi(\sigma')) \end{aligned}$$

相应地可以定义以公式为前后断言的模型的正确性。

§2.1.3 卫式迁移模型

卫式迁移模型是一阶逻辑的扩充。 卫式迁移模型所描述的迁移关系主要有两个要素：卫式和赋值。除此之外，系统还包含初始条件的描述。 在一阶逻辑的基础上，我们增加符号

$$\{\longrightarrow, :=\}$$

给定 $B = (F, P)$ 和变量集合 V 。定义 (B, V) 上的迁移如下：

$$p \longrightarrow (v_1, v_2, ..., v_n) := (e_1, e_2, ..., e_n)$$

其中 $p \in QFF_B$ 为一个不带量词的公式且 $Var(p) \subseteq V$ ， $v_1, v_2, ..., v_n \in V$ 为变元， $e_1, e_2, ..., e_n \in T_B$ 为 B 上的项且 $\bigcup_{i=1}^{n} Var(e_i) \subseteq V$ 。一个迁移表示系统的一个原子动作。

Definition 2.3 一个 (B, V) 上的卫式迁移模型是一个二元组

$$(T, \Theta)$$

其中 T 为 (B, V) 上迁移的有限集合， $\Theta \in QFF_B$ 为初始条件，是一个不带量词的公式且 $Var(\Theta) \subseteq V$ 。

B 的解释确定了迁移系统描述中常元符号、函数符号和谓词符号的含义。 变量的赋值 σ 代表了系统运行中变量在某一时刻的状态。因此也称 σ 为状态。状态的集合为 Σ 。 以下假设 B 的解释 $I = (D, I_0)$ 为已经给定。

状态空间

系统的状态空间为 V 中变量取值的组合 $\Sigma = \{\sigma | \sigma(x) \in D, x \in V\}$ 。

状态迁移关系

迁移 $\varphi \longrightarrow (v_1, v_2, ..., v_n) := (e_1, e_2, ..., e_n)$ 在状态 σ 可执行，当且仅当 $\sigma \models_I \varphi$ 。 设 t 是 $\varphi \longrightarrow (v_1, v_2, ..., v_n) := (e_1, e_2, ..., e_n)$ 为一个迁移。用 $\sigma \xrightarrow{t} \sigma'$ 表示 t 在状态 σ 可执行且 $\sigma' = \sigma[v_1/I(e_1)(\sigma)] \cdots [v_n/I(e_n)(\sigma)]$ 。

用 $\sigma \rightarrow \sigma'$ 表示

在状态 σ	存在可执行迁移 t 且 $\sigma \xrightarrow{t} \sigma'$
在状态 σ	不存在可执行迁移且 $\sigma' = \sigma$

运行

用 $[\sigma_i]_{i \geq 0}$ 表示无穷系列

$$\sigma_0 \sigma_1 \sigma_2 \sigma_3 .....$$

卫式迁移模型 (T, Θ) 上的一个运行是状态集 Σ 上的一个无穷序列 $[\sigma_i]_{i \geq 0}$ 满足 $\sigma_0 \models_I \Theta$ 且对任意 $i \geq 0$,

$$\sigma_i \rightarrow \sigma_{i+1}$$

可达状态

用 $\sigma \rightarrow \sigma'$ 表示存在 $t \in T$ 使得 $\sigma \xrightarrow{t} \sigma'$ 。状态集合 A 的可达状态集合 $rh(A)$ 为 $\{\sigma' | \sigma \xrightarrow{*} \sigma', \sigma \in A\}$ 。系统的可达状态集合 $rh(\Theta)$ 为 $\{\sigma' | \sigma \xrightarrow{*} \sigma', \sigma \models_I \Theta\}$ 。

可达性质

给定一个状态公式 φ 。 φ 状态在系统 M 中可达，当且仅当系统运行中存在满足 φ 的状态。即 $\exists \sigma \in rh(\Theta).\sigma \models_I \varphi$ 。

安全性质

安全性质为可达性质的对偶性质。给定一个状态公式 φ 。系统 M 满足安全性质 φ ，记作 $M \models \Box \varphi$ ，当且仅当系统的每个运行中的每个状态都满足 φ 。即 $\forall \sigma \in rh(\Theta).\sigma \models_I \varphi$ 。

简单响应性质

给定一个状态公式 φ 。系统 M 满足简单响应性质 φ ，记作 $M \models \Diamond \varphi$ ，当且仅当系统的每个运行中都有状态满足 φ 。 即 $\forall A \subseteq \Sigma.((\exists \sigma \in A.(\sigma \models_I \Theta)) \wedge (\forall \sigma \in A. \exists \sigma' \in A.(\sigma \rightarrow \sigma')) \rightarrow (\exists \sigma \in A.(\sigma \models_I \varphi)))$ 。

响应性质

给定状态公式 φ 和 ψ 。系统 M 满足响应性质 (φ, ψ) ，记作 $M \models \Box(\varphi \rightarrow \Diamond \psi)$ ，当且仅当系统的每个运行中每个满足 φ 的状态之后或同时有状态满足 ψ 。即 $\forall A \subseteq rh(\Theta).((\forall \sigma \in A. \exists \sigma' \in A.(\sigma \rightarrow \sigma')) \rightarrow (\exists \sigma \in A.(\sigma \models_I \varphi)))$ 。

§2.1.4 谓词迁移模型

谓词迁移模型是一阶逻辑的扩充，表示为一阶逻辑公式的二元组。

给定 $B=(F,P)$ 和变量集合 $V=\{v_1,...,v_n\}$ 。 定义 $V'=\{v'|v\in V\}$ 。

Definition 2.4 一个 (B,V) 上的谓词迁移模型是一个二元组

$$(\rho,\Theta)$$

其中 $\rho\in QFF_B$ 为迁移关系，一个不带量词的公式且 $Var(\rho)\subseteq V\cup V'$ ， $\Theta\in QFF_B$ 为初始条件，是一个不带量词的公式且 $Var(\Theta)\subseteq V$ 。

B 的解释确定了迁移系统描述中常元符号、函数符号和谓词符号的含义。 变量的赋值 σ 代表了系统运行中变量在某一时刻的状态。因此也称 σ 为状态。状态的集合为 Σ 。 以下假设 B 的解释 $I=(D,I_0)$ 为已经给定。

状态空间

系统的状态空间为 V 中变量取值的组合 $\Sigma=\{\sigma|\sigma(x)\in D,x\in V\}$ 。

状态迁移关系

迁移 ρ 在状态 σ 可执行，当且仅当存在 $a_1,...,a_n$ 使得 $\sigma[v'_1/a_1]\cdots[v'_n/a_n]\models_I\rho$ 。 用 $\sigma\rightarrow\sigma'$ 表示 存在 $a_1,...,a_n$ 满足 $\sigma[v'_1/a_1]\cdots[v'_n/a_n]\models_I\rho$ 且 $\sigma'=\sigma[v_1/a_1]\cdots[v_n/a_n]$ ，或 ρ 在状态 σ 不可执行且 $\sigma'=\sigma$ 。

运行

用 $[\sigma_i]_{i\geq 0}$ 表示无穷系列

$$\sigma_0\sigma_1\sigma_2\sigma_3.....$$

谓词迁移模型 (ρ,Θ) 上的一个运行是状态集 Σ 上的一个无穷序列 $[\sigma_i]_{i\geq 0}$ 满足 $\sigma_0\models_I\Theta$ 且对任意 $i\geq 0$,

$$\sigma_i\rightarrow\sigma_{i+1}$$

可达状态

用 $\sigma\rightarrow\sigma'$ 表示存在 $t\in T$ 使得 $\sigma\xrightarrow{t}\sigma'$ 。状态集合 A 的可达状态集合 $rh(A)$ 为 $\{\sigma'|\sigma\xrightarrow{*}\sigma',\sigma\in A\}$ 。系统的可达状态集合 $rh(\Theta)$ 为 $\{\sigma'|\sigma\xrightarrow{*}\sigma',\sigma\models_I\Theta\}$ 。

安全性质和响应性质

类似于条件迁移模型，我们可以讨论安全性质和响应性质。

§2.1.5 不同模型的关系

以上介绍了结构化循环语句模型、流程图模型、卫式迁移模型和逻辑迁移模型。有些模型可以相互转换。

给定结构化循环语句模型 S 。我们可以构造一个相应的流程图模型。 我们用 S 构造带标号的程序。这个构造记为 f ，定义如下

S	$f(S)$
ϵ	end
$x := t$	$x := t$
$T_1; T_2$	$f(T_1);\quad l_{T_2} : f(T_2)$
if e then T_1 else T_2 fi	if e then $l_{T_1} : f(T_1)$ else $l_{T_2} : f(T_2)$ fi
while e do T od	while e do $l_T : f(T)$ od

我们用带标号的程序 S 构造流程图模型。这个构造记为 g，定义如下（其中 $l(S)$ 为 S 的标号部分）：

S	$g(S)$	
end	$\{\}$	
l_T	$\{\}$	
l_T : x:=t; T_1	$\{l_T : x := t$ goto $l(T_1);\}$	$\cup\, g(T_1)$
l_T : if e then T_1 else T_2 fi; T_3	$\{l_T$: if (e) goto $l(T_1)$ else goto $l(T_2);\}$	$\cup\, g(T_1; l(T_3)) \cup g(T_2; l(T_3)) \cup g(T_3)$
l_T : while e do T_1 od; T_2	$\{l_T$: if (e) goto $l(T_1)$ else goto $l(T_2);\}$	$\cup\, g(T_1; l_T) \cup g(T_2)$

给定结构化循环语句模型 S。则 $g(beg : f(S))$ 为相应的流程图模型。

谓结构化循环语句模型和流程图模型的等价

给定 (B, V) 上的结构化循环语句模型 S 和 (B, V) 上的流程图模型 T。如果对任意给定的 B 的解释，对 S 的任意一个运行 $[(S_i, \sigma_i)]_{i\geq 0}$ 存在 T 的一个运行 $[(l_i, \sigma'_i)]_{i\geq 0}$ 满足 $\sigma_i = \sigma'_i$ 且 $S_i = \epsilon$ 当且仅当 $l_i = end$，反之亦然，则称结构化循环语句模型 S 和流程图模型 T 等价。

构造正确性

给定结构化循环语句模型 S。则流程图模型 $g(beg : f(S))$ 与 S 等价。

§2.2 显式状态迁移模型

在隐式迁移模型的描述中，系统运行的状态空间以及系统的运行由解释 I 所决定。明确的状态空间以及状态之间迁移关系的描述有助于简化系统性质的检验。

§2.2.1 Kripke 模型 (KS)

以条件迁移模型或谓词迁移模型为出发点，状态空间即是 Σ。除了状态空间之外，描述系统运行的基本要素还有状态之间的迁移关系，以及系统的初始状态。

Definition 2.5 一个 *Kripke* 模型是一个三元组

$$\langle S, \Delta, I\rangle$$

其中 S 为状态集合，$\Delta \subseteq S \times S$ 为 S 上的完全迁移关系，$I \subseteq S$ 为初始状态集。

路径和运行

用 $s \to s'$ 表示存在从 s 到 s' 的迁移，即 $(s, s') \in \Delta$。用 $[s_i]_{i=0}^n$ 表示有穷序列

$$s_0 s_1 s_2 ... s_n$$

Kripke 模型上的有穷路径是 S 上的有穷序列 $[s_i]_{i=0}^n$ 满足对任意 $0 \leq i \leq n-1$, $s_i \to s_{i+1}$。
Kripke 模型上的无穷路径是 S 上的无穷序列 $[s_i]_{i\geq 0}$ 满足对任意 $i \geq 0$, $s_i \to s_{i+1}$。
Kripke 模型上的运行是 Kripke 模型上的无穷路径 $[s_i]_{i\geq 0}$ 满足 $s_0 \in I$。

可达状态

用 $\xrightarrow{*}$ 表示 $\rightarrow$ 的传递自反闭包。状态集合 A 的可达状态集合 $rh(A)$ 为 $\{s'|s\xrightarrow{*}s', s\in A\}$ 。系统的可达状态集合为 $rh(I)$ 。

安全性质

系统的安全性质可用状态集合表示。状态集合 A 安全，当且仅当系统的每个运行中的每个状态都在 A 中。即 $rh(I)\subseteq A$ 。

简单响应性质

系统的响应性质可用状态集合表示。状态集合 A 是简单响应的，当且仅当系统的每个运行中有状态在 A 中。 即 $\forall B\subseteq S.((B\cap rh(I)\neq\emptyset)\wedge(\forall s\in B.\exists s'\in B.(s\rightarrow s'))\rightarrow(B\cap A\neq\emptyset))$ 。

§2.2.2 标号 KS

由于 KS 对于系统的描述过于简化， KS 作为软件模型，用状态集表示性质显得很复杂。如果表示性质的集合很大，则其检查就很复杂。为了表达和计算的简单，可用命题表示性质。为了能够用命题表示性质，我们必须将结构中的状态和什么公式在哪些状态上成立联系起来。为了定义上的简单，我们只考虑在每个状态上某些简单命题是否成立。

Definition 2.6 *给定一个原子命题集合 AP 。一个 AP 上的标号 KS 是一个四元组*

$$\langle S,\Delta,I,L\rangle$$

其中 $\langle S,\Delta,I\rangle$ 为 Kripke 模型， $L:S\rightarrow 2^{AP}$ 为状态集合到 AP 的幂集的映射。

在这样的一个结构中， $L(s)$ 表示在 s 上所有 s 成立的原子命题的集合。原子命题 p 在 s 上成立当且仅当 $p\in L(s)$ ，其它命题逻辑公式的成立与否遵从命题逻辑的解释。

记原子命题集合 AP 上所有命题公式的集合为 $\mathcal{L}_{AP}$ 。我们可以用命题逻辑的公式表示性质。给定一个 AP 上的标号 Kripke 模型 $\langle S,\Delta,I,L\rangle$ ，一个状态 $s\in S$ 满足性质 $\varphi\in\mathcal{L}_{AP}$ 记作 $M,s\models\varphi$ ，其定义如下： $M,s\models\varphi$ 成立当且仅当：

$C,s\models p$	若 $p\in AP$ 且 $p\in L(s)$ 。
$M,s\models\neg\psi$	若 $M,s\not\models\psi$ 。
$M,s\models\psi_0\vee\psi_1$	若 $M,s\models\psi_0$ 或 $M,s\models\psi_1$ 。
$M,s\models\psi_0\wedge\psi_1$	若 $M,s\models\psi_0$ 且 $M,s\models\psi_1$ 。
$M,s\models\psi_0\rightarrow\psi_1$	若 $M,s\models\psi_0$ 则 $M,s\models\psi_1$ 。
$M,s\models\psi_0\leftrightarrow\psi_1$	若 $M,s\models\psi_0$ 当且仅当 $M,s\models\psi_1$ 。

设 $m=\{x_1,...,x_k\}\subseteq AP$ 且 $AP\setminus m=\{y_1,...,y_l\}$ 。定义 $m\models\varphi$ 当且仅当 $(\varphi^{1,...,1}_{x_1,...,x_k})^{0,...,0}_{y_1,...,y_l}$ 的值为 1 。则 $M,s\models\varphi$ 当且仅当 $L(s)\models\varphi$ 。

语言

M 的语言记为 $\mathcal{L}(M)\subseteq(2^{AP})^\omega$ ，定义如下： $u=u_0u_1\cdots\in\mathcal{L}(M)$ 当且仅当 M 有一个运行 $r=r_0r_1\cdots$ 且对所有 i ， $u_i=L(r_i)$ 。

公式与状态集合

一个公式对应于一个状态集合。即 φ 对应于状态集合 $\{s|M,s\models\varphi\}$ 。用 $[[\varphi]]$ 表示状态集合 $\{s|M,s\models\varphi\}$ 。

安全性质

系统的安全性质可用公式表示。系统满足安全性质 φ，当且仅当系统的每个运行中的每个状态都满足 φ。即 $\forall s \in rh(I).(M, s \models \varphi)$，即 $\forall s \in rh(I).(L(s) \models \varphi)$，即 $rh(I) \subseteq [[\varphi]]$。

简单响应性质

系统的响应性质可用公式表示。系统满足响应性质 φ，当且仅当系统的每个运行中都有状态都满足 φ。即 $\forall B \subseteq S.((B \cap rh(I) \neq \emptyset) \wedge (\forall s \in B.\exists s' \in B.(s \to s')) \to (\exists s \in B.(L(s) \models \varphi))$。

§2.2.3 扩展标号 KS

标号 KS 只考虑在每个状态上某些简单命题是否成立，我们可以对其进行扩展，在每个状态上标上某个公式是否成立。

Definition 2.7 *给定一个原子命题集合 AP。一个 AP 上的扩展标号 KS 是一个四元组*

$$\langle S, \Delta, I, L \rangle$$

其中 $\langle S, \Delta, I \rangle$ 为 Kripke 模型，$L : S \to \mathcal{L}_{AP}$ 为状态集合到命题逻辑公式集合的映射。

给定一个 AP 上的扩展标号 Kripke 模型 $\langle S, \Delta, I, L \rangle$，一个状态 $s \in S$ 满足性质 $\varphi \in \mathcal{L}_{AP}$ 记作 $M, s \models \varphi$ 当且仅当 $L(s) \to \varphi$。

语言

M 的语言记为 $\mathcal{L}(M) \subseteq (\mathcal{L}_{AP})^\omega$，定义如下：$u = u_0 u_1 \cdots \in \mathcal{L}(M)$ 当且仅当 M 有一个运行 $r = r_0 r_1 \cdots$ 且对所有 i，$u_i \models L(r_i)$。

安全性质

系统满足安全性质 $\varphi \in \mathcal{L}_{AP}$，当且仅当系统的每个运行中的每个状态都满足 φ，即

$$\forall s \in rh(I).(L(s) \to \varphi)。$$

简单响应性质

系统的响应性质可用公式表示。系统满足响应性质 φ，当且仅当系统的每个运行中都有状态都满足 φ。即

$$\forall B \subseteq S.((B \cap rh(I) \neq \emptyset) \wedge (\forall s \in B.\exists s' \in B.(s \to s')) \to \exists s \in B.(L(s) \wedge \varphi \text{是可满足的}))。$$

§2.2.4 公平标号 Kripke 模型

对于 Kripke 模型而言，只要满足状态迁移关系的无穷状态序列就是系统的一个运行。有时候为了更精确地描述系统的运行，我们需要对运行做一定限制以排除一些满足状态转换关系但不合理的无穷状态序列。 若要求能够排除这样的无穷状态序列，我们需要在结构中加入新的成分，即设置公平运行的条件，定义公平标号 Kripke 模型。

Definition 2.8 *一个公平标号 Kripke 模型是一个五元组*

$$\langle S, \Delta, I, L, F \rangle$$

其中 $\langle S, \Delta, I, L \rangle$ 是一个 AP 上的标号 Kripke 模型，$F \subseteq 2^S$ 为公平性约束的集合。

运行

公平标号 Kripke 模型 $\langle S, \Delta, I, L, F\rangle$ 上的一个运行即 Kripke 模型 $\langle S, \Delta, I\rangle$ 上的一个运行。 定义 $inf(\pi)$ 为无限多次出现在 π 中的状态的集合。一个公平标号 Kripke 模型上的运行 $\pi = [s_i]_{i\geq 0}$ 是公平的，当且仅当对所有 $f \in F$ ，

$$inf(\pi) \cap f \neq \emptyset$$

语言

M 的语言记为 $\mathcal{L}(M) \subseteq (2^{AP})^\omega$ ，定义如下： $u = u_0u_1 \cdots \in \mathcal{L}(M)$ 当且仅当 M 有一个公平运行 $r = r_0r_1 \cdots$ 且对所有 i ， $u_i \models L(s_i)$ 。

语言非空性质

给定 $M = \langle S, \Delta, I, L, F\rangle$ 。 $\mathcal{L}(M) \neq \emptyset$ 当且仅当 M 有一个公平运行 当且仅当有向图 (S, Δ) 存在由 I 可达的强连通分量 C 且对所有 $f \in F$ ， $C \cap f \neq \emptyset$ 。

§2.2.5 不同模型的关系

以上介绍了 Kripke 模型、标号 Kripke 模型、扩展标号 Kripke 模型和公平标号 Kripke 模型。部分模型之间可以相互转换。部分模型可以由隐式迁移系统生成。

标号 KS 与扩展标号 KS

给定 AP 上的扩展标号 KS 模型 $M = \langle S, \Delta, I, L\rangle$ 。我们可以构造一个相应的标号 KS 模型。 定义 $M' = \langle S', \Delta', I', L'\rangle$ 如下

$S' =$	$\{(s, m) \mid s \in S, m \models L(s)\}$
$\Delta' =$	$\{((s, m), (s', m')) \mid \{(s, s') \in \Delta\}$
$I' =$	$\{(s, m) \mid \{s \in I\}$
$L' :$	$L((s, m)) = m$

则 $\mathcal{L}(M') = \mathcal{L}(M)$ 。

谓词迁移模型与标号 KS

KS 可由有穷状态的谓词迁移模型或卫式迁移模型生成。若 KS 由谓词迁移模型生成， KS 的状态空间的大小由该模型的变量及其取值范围决定。给定 (B, V) 上的谓词迁移模型 (ρ, Θ) 。给定 B 上的解释 $\mathcal{I}$ 。设 $V = \{v_1, ..., v_n\}$ 且令 $|v|$ 为 v 可能取值的个数。

状态空间的大小为 $\lvert v_1\rvert \cdot \lvert v_2\rvert \cdots \lvert v_n\rvert$ 。 若用 n 元组表示则每个状态为 $\langle a_1, ..., a_n\rangle$ 其中 $a_1, ..., a_n$ 分别为 $v_1, ..., v_n$ 的值。
迁移集合由 ρ 产生。 设 $\sigma, \sigma' \in \Sigma$ 。若 $\sigma \rightarrow \sigma'$ ，则 $(\langle\sigma(v_1), \sigma(v_2), ..., \sigma(v_n)\rangle, \langle\sigma'(v_1), \sigma'(v_2), ..., \sigma'(v_n)\rangle) \in \Delta$ 。
初始状态集合 I 根据 Θ 定义产生。若 $\sigma \models_{\mathcal{I}} \Theta$ 为真，则 $\langle\sigma(v_1), \sigma(v_2), ..., \sigma(v_n)\rangle \in I$ 。

对每个变量 v ，设其值域为 $\{a_1, ..., a_{|v|}\}$ 。我们可以构造 $|v|$ 个命题 $\{p_1, ..., p_{|v|}\}$ ，每个命题代表 $v = a_1, v = a_2, ..., v = a_{|v|}$ 中对应的一项。 设 $p_{v,a}$ 代表 $v = a$ ，定义 L 如下： $p_{x_i,a} \in L(\langle x_1, ..., x_n\rangle)$ 当且仅当 $x_i = a$ 。这样，我们有一个原子命题集合 AP 和其上的标号 Kripke 模型 $M_{(\rho,\Theta)} = \langle S, \Delta, I, L\rangle$ 。

谓词迁移模型与标号 KS 的等价

给定 (B,V) 上的谓词迁移模型 (ρ,Θ) 和 B 上的解释 $\mathcal{I}=(D,\mathcal{I}_0)$ 其中 D 为有穷集合。给定 AP 上的标号 KS 模型 $M=\langle S,\Delta,I,L\rangle$。给定一个 AP 子集到谓词集合 $\{v=d|v\in V,d\in D\}$ 子集的一一对应关系 ζ。 如果对任意 (ρ,Θ) 的一个运行 $\sigma_0\sigma_1\cdots$ 存在 M 的一个运行 $s_0s_1\cdots$ 满足 $\sigma_i\models(v=d)$ 当且仅当 $\zeta^{-1}(v=d)\in L(s_i)$，且对任意 M 的一个运行 $s_0s_1\cdots$ 存在 (ρ,Θ) 的一个运行 $\sigma_0\sigma_1\cdots$ 满足 $p\in L(s_i)$ 当且仅当 $\sigma_i\models\zeta(p)$，则称谓词迁移模型与标号 KS 是 ζ 等价的。

构造正确性

给定 (B,V) 上的谓词迁移模型 (ρ,Θ) 和 B 上的解释 $\mathcal{I}=(D,\mathcal{I}_0)$ 其中 D 为有穷集合。给定 $\zeta=\{(p_{x,a},(x=a))|x\in V,a\in D\}$。则谓词迁移模型 (ρ,Θ) 与标号 KS 模型 $M_{(\rho,\Theta)}$ 是 ζ 等价的。

标号 KS 上的模拟与互模拟关系

设 $M_1=\langle S_1,\Delta_1,I_1,L_1\rangle$ 和 $M_2=\langle S_2,\Delta_2,I_2,L_2\rangle$ 为原子命题集合 AP 上的标号 KS。

$\sigma\subseteq S_1\times S_2$ 是 M_1 到 M_2 的模拟关系当且仅当 σ 是有向图 $\langle S_1,\Delta_1\rangle$ 到 $\langle S_2,\Delta_2\rangle$ 的模拟关系且 σ 满足 （1）对所有 $(x,y)\in S_1\times S_2$，若 $(x,y)\in\sigma$ 则 $L_1(x)=L_2(y)$；（2）对所有 $x\in I_1$，存在 $y\in I_2$ 满足 $(x,y)\in\sigma$。 若存在一个这样的 σ，则称 M_2 模拟 M_1。

$\sigma\subseteq S_1\times S_2$ 是 M_1 和 M_2 上的互模拟关系当且仅当 σ 是有向图 $\langle S_1,\Delta_1\rangle$ 和 $\langle S_2,\Delta_2\rangle$ 上的互模拟关系且 σ 满足 （1）对所有 $(x,y)\in S_1\times S_2$，若 $(x,y)\in\sigma$ 则 $L_1(x)=L_2(y)$；（2）对所有 $x\in I_1$，存在 $y\in I_2$ 满足 $(x,y)\in\sigma$，且对所有 $y\in I_2$，存在 $x\in I_1$ 满足 $(x,y)\in\sigma$。若存在一个这样的 σ，则称 M_2 与 M_1 互模拟。

§2.3 标号迁移系统与自动机

Kripke 模型描述的主要是状态和状态之间的转换关系，不关心是什么动作使得状态发生了变化。在某些时候动作也是系统描述中关键的一部分。

§2.3.1 标号迁移系统 (LTS)

Definition 2.9 一个标号迁移系统是一个四元组

$$\langle\Sigma,S,\Delta,I\rangle$$

其中 Σ 为标号集合，S 为状态集合，$\Delta\subseteq S\times\Sigma\times S$ 是一个有标号的迁移关系，I 为初始状态集。

运行

用 $s\xrightarrow{a}s'$ 表示存在从 s 到 s' 的 a 迁移，即 $(s,a,s')\in\Delta$。LTS 上的一个运行是状态集 S 上的一个无穷序列 $[s_i]_{i\geq0}$ 满足 $s_0\in I$ 且对任意 $i\geq0$，存在 $a\in\Sigma$，使得 $s_i\xrightarrow{a}s_{i+1}$。

字符串

标号集 Σ 上的一个字符串是 Σ 上的一个无穷序列。$[a_i]_{i\geq1}\in\Sigma^\omega$ 是标号迁移系统 $M=\langle\Sigma,S,\Delta,I\rangle$ 上的一个字符串当且仅当存在 LTS 上的一个运行 $\pi=[s_i]_{i\geq0}$ 使得对任意 $i\geq0$，$s_i\xrightarrow{a_{i+1}}s_{i+1}$。$\pi$ 称为字符串 ω 上的一个运行。

语言

标号迁移系统 M 上的字符串的集合是 M 的语言，记为 $\mathcal{L}(M)$。

双标号迁移系统

与 KS 类似，我们可以在 LTS 的状态上记载一些信息。

Definition 2.10 给定一个原子命题集合 AP 。一个 AP 上的标号迁移结构是一个五元组

$$\langle \Sigma, S, \Delta, I, L \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号迁移系统， $L : S \to 2^{AP}$ 为状态集到 AP 的幂集的映射。

§2.3.2 ω 自动机

对于 LTS 而言，只要满足状态迁移关系的无穷状态序列就是系统的一个运行。有时候为了更精确地描述系统的运行，我们需要对运行做一定限制以排除一些满足状态转换关系但不合理的无穷状态序列。 若要求能够排除这样的无穷状态序列，我们需要在结构中加入新的成分。我们在 LTS 的基础上，为其运行设置可接受条件，定义 ω 自动机。

Büchi 自动机

为简单起见，我们先定义只有一个简单接受条件的 Büchi 自动机。

Definition 2.11 一个 *Büchi* 自动机是一个五元组

$$\langle \Sigma, S, \Delta, I, F \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号迁移系统， $F \subseteq S$ 为自动机的接受状态集。

可接受运行

Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 上的运行。 $\mathcal{B}$ 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的当且仅当

$$inf(\pi) \cap F \neq \emptyset$$

语言

$\mathcal{B}$ 的标号集 Σ 上的一个字符串 $\omega = [a_i]_{i \geq 1}$ 是可接受的当且仅当存在 ω 上的可接受运行。$\mathcal{B}$ 上可接受字符串的集合为 $\mathcal{B}$ 的语言，记作 $\mathcal{L}(\mathcal{B})$ 。

自动机的运算

自动机的运算包括并、交和补。 A' 是 A 和 B 的并自动机，记作 $A = A \cup B$ ，当前仅当 $\mathcal{L}(A') = \mathcal{L}(A) \cup \mathcal{L}(B)$ 。 A' 是 A 和 B 的交自动机，记作 $A = A \cap B$ ，当前仅当 $\mathcal{L}(A') = \mathcal{L}(A) \cap \mathcal{L}(B)$ 。 A' 是 A 的补自动机，记作 $A' = \neg A$ ，当前仅当 $\mathcal{L}(A') = \Sigma^\omega \setminus \mathcal{L}(A)$ 。

自 Büchi 自动机的并

给定两个 Büchi 自动机 $\mathcal{B}_1 = \langle \Sigma, S_1, \Delta_1, I_1, F_1 \rangle$, $\mathcal{B}_2 = \langle \Sigma, S_2, \Delta_2, I_2, F_2 \rangle$ 且 $S_1 \cap S_2 = \emptyset$ 。
定义 $\mathcal{B}_1 \cup \mathcal{B}_2 = \langle \Sigma, S_1 \cup S_2, \Delta_1 \cup \Delta_2, I_1 \cup I_2, F_1 \cup F_2 \rangle$ ，则 $\mathcal{L}(\mathcal{B}_1 \cup \mathcal{B}_2) = \mathcal{L}(\mathcal{B}_1) \cup \mathcal{L}(\mathcal{B}_2)$ 。

扩展 Büchi 自动机

为了能够表示多个接受条件，我们扩展 Büchi 自动机的定义，将接受状态集合改为接受状态集的集合。

Definition 2.12 *一个扩展 Büchi 自动机是一个五元组*

$$\langle \Sigma, S, \Delta, I, F \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号迁移系统， $F \subseteq 2^S$ 为自动机的接受状态集的集合。

可接受运行

扩展 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 上的运行。$\mathcal{B}$ 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的，当且仅当对所有 $f \in F$ ，

$$inf(\pi) \cap f \neq \emptyset$$

自动机的等价

一个自动机等价于另一个自动机当且仅当其所定义的语言是相同的。两类自动机的表达能力相同，当且仅当任意一类自动机的一个实例都有一个等价的另一类自动机的一个实例。 扩展 Büchi 自动机的表达能力和 Büchi 自动机是一样的。

自扩展 Büchi 自动机的并和交

给定两个扩展 Büchi 自动机 $\mathcal{B}_1 = \langle \Sigma, S_1, \Delta_1, I_1, F_1 \rangle, \mathcal{B}_2 = \langle \Sigma, S_2, \Delta_2, I_2, F_2 \rangle$ 且 $S_1 \cap S_2 = \emptyset$ 。
设

$$F = \{f \cup S_2 | f \in F_1\} \cup \{f \cup S_1 | f \in F_2\}$$

定义 $\mathcal{B}_1 \cup \mathcal{B}_2 = \langle \Sigma, S_1 \cup S_2, \Delta_1 \cup \Delta_2, I_1 \cup I_2, F \rangle$ ，则 $\mathcal{L}(\mathcal{B}_1 \cup \mathcal{B}_2) = \mathcal{L}(\mathcal{B}_1) \cup \mathcal{L}(\mathcal{B}_2)$ 。
设

$$\Delta = \{((q_1, q_2), a, (q_1', q_2')) | (q_1, a, q_1') \in \Delta_1, (q_2, a, q_2') \in \Delta_2\}$$
$$F = \{f \times S_2 | f \in F_1\} \cup \{S_1 \times f | f \in F_2\}$$

定义 $\mathcal{B}_1 \cap \mathcal{B}_2 = \langle \Sigma, S_1 \times S_2, \Delta, I_1 \times I_2, F \rangle$ ，则 $\mathcal{L}(\mathcal{B}_1 \cap \mathcal{B}_2) = \mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$ 。

扩展 Büchi 自动机是 Büchi 自动机的一种扩展，在扩展 Büchi 自动机基础上再对其接受条件做进一步扩展或重新定义，我们可以得到 Streett 自动机、 Rabin 自动机或 Muller 自动机。

Streett 自动机

Definition 2.13 *一个 Streett 自动机是一个五元组*

$$\langle \Sigma, S, \Delta, I, F \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号迁移系统， $F \subseteq 2^S \times 2^S$ 为自动机的接受状态集的集合。

可接受运行

设 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 是 Streett 自动机。 $\mathcal{B}$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 上的运行。 $\mathcal{B}$ 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的，当且仅当对所有 $(f, g) \in F$ ，

$$inf(\pi) \cap f \neq \emptyset \rightarrow inf(\pi) \cap g \neq \emptyset$$

Rabin 自动机

Rabin 自动机的结构和 Streett 自动机完全一样，只是接受条件正好和 Streett 自动机的接受条件相反。给定一个 Rabin 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 。 $\mathcal{B}$ 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的，当且仅当存在 $(f, g) \in F$ ，

$$inf(\pi) \cap f \neq \emptyset \wedge inf(\pi) \cap g = \emptyset$$

Muller 自动机

Definition 2.14 *一个 Muller 自动机是一个五元组*

$$\langle \Sigma, S, \Delta, I, F \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 为迁移系统， $F \subseteq 2^S$ 为自动机的接受状态集的集合。

可接受运行

设 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 是 Muller 自动机。 $\mathcal{B}$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 上的运行。 $\mathcal{B}$ 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的，当且仅当

$$inf(\pi) \in F$$

Muller 自动机的补

给定 Muller 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 。
定义 $\neg\mathcal{B} = \langle \Sigma, S, \Delta, I, 2^S \setminus F \rangle$ ，则 $\mathcal{L}(\neg\mathcal{B}) = (\Sigma)^\omega \setminus \mathcal{L}(\mathcal{B})$ 。

表达能力

从语言的角度，Büchi 自动机、扩展 Büchi 自动机、Streett 自动机、Rabin 自动机、Muller 自动机与基于迁移的 Büchi 扩展自动机的表达能力相同，且都是补封闭的。

语言非空与语言包含

$\mathcal{L}(A) \subseteq \mathcal{L}(B)$ 当且仅当 $\mathcal{L}(A) \cap (\Sigma^\omega \setminus \mathcal{L}(B)) = \emptyset$ 当且仅当 $\mathcal{L}(A \cap \neg B) = \emptyset$ 。

基于迁移的扩展 Büchi 自动机

以上自动机的接受条件都是状态集合。我们也可以将接受条件定义在迁移上。以下定义基于迁移的扩展 Büchi 自动机。

Definition 2.15 *基于迁移的扩展 Büchi 自动机是一个五元组*

$$\langle S, \Sigma, \Delta, I, F \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号迁移系统， $F \subseteq 2^\Delta$ 是接受迁移集合。

字符串上的运行

$[s_i]_{i \geq 0}$ 是字符串 $[a_i]_{i \geq 1}$ 上的一个运行，当且仅当对所有 $i \geq 0$ ， $(s_i, a_{i+1}, s_{i+1}) \in \Delta$ 。

可接受字运行

用 $[(s_i, a_{i+1}, s_{i+1})]_{i\geq 0}$ 表示

$$(s_0, a_1, s_1)(s_1, a_2, s_2)(s_2, a_3, s_3)\cdots$$

定义 $inf([(s_i, a_{i+1}, s_{i+1})]_{i\geq 0})$ 为无限多次出现在 $[(s_i, a_{i+1}, s_{i+1})]_{i\geq 0}$ 中的迁移的集合。
字符串 $[a_i]_{i\geq 1}$ 上的一个运行 $[s_i]_{i\geq 0}$ 是可接受的，当且仅当对所有 $f \in F$ ，

$$inf([(s_i, a_{i+1}, s_{i+1})]_{i\geq 0}) \cap f \neq \emptyset$$

表达能力

字母表 Σ 上的一个字符串 ω 是可接受的当且仅当存在 ω 上的可接受运行。基于迁移的 Büchi 自动机与 Büchi 自动机的表达能力相同。

§2.3.3 标号交错迁移系统与交错自动机

标号迁移系统的每个动作其后续状态可能有多个，但每次运行只能从其中选择一个状态。对这类迁移系统的一种扩展是让其可以选择多个后续状态。这样扩展的迁移系统的一次运行可以是树状结构。

标号交错迁移系统 (ATS)

Definition 2.16 一个标号交错迁移系统是一个四元组 $\langle \Sigma, S, \Delta, I\rangle$ ，其中 Σ 为动作集合， S 为状态集合， $\Delta \subseteq S \times \Sigma \times 2^S$ 为标号的迁移关系， I 为系统初始状态的集合。

字符串上的运行

给定一个 Σ 上的无限长的字符串 $\omega = a_0a_1a_2...$ 。标号交错迁移系统在 ω 上的一个运行是状态集 S 上的一颗树 r 。设 $r(0)$ 为 r 的根节点的单点集、 $r(i)$ 为 r 的第 i 层节点的集合、 $child(x)$ 为节点 x 的子节点的集合。用 $\Delta(x, a)$ 表示集合 $\{y|(x, a, y) \in \Delta\}$ 。 r 满足以下条件。

$r(0) \subseteq I$
若 $x \in r(i)$ 为第 i 层的一个节点，则 $child(x) \in \Delta(x, a_i)$

若 $\Delta(x, a_i)$ 包含空集，则 x 可以没有子节点。

运行

状态集 S 上的一颗树 r 是标号交错迁移系统 $\langle \Sigma, S, \Delta, I\rangle$ 的一个运行当且仅当存在一个 Σ 上的无限长的字符串 $\omega = a_0a_1a_2...$ 使得 r 是标号交错迁移系统在 ω 上的一个运行。

§2.3.4 双标号交错迁移系统

与 KS 类似，我们可以在 ATS 的状态上记载一些信息。

Definition 2.17 给定一个原子命题集合 AP 。一个 AP 上的双标号交错迁移系统是一个五元组

$$\langle \Sigma, S, \Delta, I, L\rangle$$

其中 $\langle \Sigma, S, \Delta, I\rangle$ 是一个标号交错迁移系统， $L : S \to 2^{AP}$ 为状态集到 AP 的幂集的映射。

交错 Büchi 自动机

Definition 2.18 一个交错 *Büchi* 自动机是一个五元组

$$\langle \Sigma, S, \Delta, I, F \rangle$$

其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号交错迁移系统， $F \subseteq S$ 为自动机的接受状态集。

可接受运行

交错 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 是上的运行就是标号交错迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 上的运行。 $\mathcal{B}$ 的一个运行树 r 是可接受的，当且仅当 r 的所有无穷路径 ρ 满足

$$inf(\rho) \cap F \neq \emptyset$$

表达能力

字母表 Σ 上的一个字符串 ω 是可接受的当且仅当存在 ω 上的可接受运行树。交错 Büchi 自动机与 Büchi 自动机的表达能力相同。

§2.3.5 确定型自动机与非确定型自动机

ω- 自动机由迁移系统和接受条件组成。不同的接受条件定义了不同的自动机。我们也可以对迁移系统分类，得到确定型自动机和非确定型自动机。确定型自动机的初始状态只有一个且其迁移关系受以下限制：

$$(s, a, s'), (s, a, s'') \in \Delta \rightarrow s' = s''$$

给定一个字符串。对确定型自动机来讲，若有与之匹配的运行，则这样的运行是唯一的。

表达能力

确定型 Muller 自动机是补封闭的，其表达能力与非确定型 Muller 自动机相同。确定型 Büchi 自动机不是补封闭的。

§2.3.6 不同模型的关系

以上介绍了不同类型的自动机。部分自动机模型等价于公平标号 Kripke 模型。以上的非确定型自动机都可以互相转换并保持语言等价。

公平标号 Kripke 模型与扩展 Büchi 自动机

给定 AP 上的公平标号 Kripke 模型 $\mathcal{M} = \langle S, \Delta, I, L, F \rangle$ 。定义

Σ	$=$	2^{AP}
Δ'	$=$	$\{(s, a, s) \mid (s, s') \in \Delta, a = L(s)\}$

定义扩展 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta', I, F \rangle$ ， 则 $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{M})$ 。

扩展 Büchi 自动机与 Büchi 自动机

给定扩展 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, \{f_1, ..., f_n\} \rangle$ 。定义

$n(s,i)$	$=$	if $(i=n)$ then 0; else if $(s \in f_{i+1})$ then $i+1$; else i.
S'	$=$	$S \times \{0,...,n\}$
Δ'	$=$	$\{(s,i),a,(s',n(s',i))\|(s,a,s') \in \Delta, i \in \{0,...,n\}\}$
I'	$=$	$I \times \{0\}$
F'	$=$	$S \times \{n\}$

定义 Büchi 自动机 $\mathcal{B}' = \langle \Sigma, S', \Delta', I', F' \rangle$ ，则 $\mathcal{L}(\mathcal{B}') = \mathcal{L}(\mathcal{B})$ 。

基于迁移的扩展 Büchi 自动机与 Büchi 自动机

给定基于迁移的扩展 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ ，定义

$n(s,a,s',i)$	$=$	if $(i=n)$ then 0; else if $((s,a,s') \in f_{i+1})$ then $i+1$; else i.
S'	$=$	$S \times \{0,...,n\}$
Δ'	$=$	$\{(s,i),a,(s',n(s,a,s',i))\|(s,a,s') \in \Delta, i \in \{0,...,n\}\}$
I'	$=$	$I \times \{0\}$
F'	$=$	$S \times \{n\}$

定义 Büchi 自动机 $\mathcal{B}' = \langle \Sigma, S', \Delta', I', F' \rangle$ ，则 $\mathcal{L}(\mathcal{B}') = \mathcal{L}(\mathcal{B})$ 。 由 Büchi 自动机，也可构造语言与之相同的基于迁移的扩展 Büchi 自动机。因此基于迁移的扩展 Büchi 自动机的表达能力和 Büchi 自动机是一样的。

§2.4 时间迁移系统和混成系统

对一些系统，我们除了关系系统的控制状态外，可能还关心在一个状态持续的时间里或两个状态之间的一些量变化。

§2.4.1 时间迁移系统 (TTS)

为了能够描述迁移动作之间时间长短的限制，我们需要在模型中加入时钟的概念。 时间的限制由时间变量上的公式表示。设 X 为时钟变量的集合， Q 为时间常量的集合。时钟公式的集合 $\Phi(X)$ 由以下语法给出。

$$\phi ::= x \leq c | c \leq x | \neg\phi | \phi \wedge \phi$$

其中 $x \in X$ 为时钟变量， $c \in Q$ 为时间常量。一个时钟赋值 v 是一个 X 到 R 的函数。 用 $v+t$ 表示对所有 $x \in X$ 满足 $v'(x) = v(x)+t$ 的时钟赋值 v' ； 用 $t \cdot v$ 表示对所有 $x \in X$ 满足 $v'(x) = t \cdot v(x)$ 的时钟赋值 v' ； 用 $[Y \to t]v$ 表示对所有 $x \in X \setminus Y$ 满足 $v'(x) = v(x)$ 和对所有 $x \in Y$ 满足 $v'(x) = t$ 的时钟赋值 v' 。

Definition 2.19 一个时间迁移系统是一个五元组

$$\langle \Sigma, S, X, \Delta, I \rangle$$

其中 Σ 为字母表， S 为状态集合， X 为时钟变量集合， $\Delta \subseteq S \times \Sigma \times 2^X \times \Phi(X) \times S$ 为迁移关系， $I \in S$ 为系统初始状态的集合。

系统状态

记 $V = X \to R$ 为时钟赋值的集合。一个系统状态为 $S \times V$ 的一个元素。

时间字符串上的运行

一个时间字符串为 $\Sigma \times R$ 上的无穷序列。为方便起见，也写成 $(\sigma, \tau) = ([\sigma_i]_{i\geq 1}, [\tau_i]_{i\geq 1}) \in \Sigma^\omega \times R^\omega$ 其中其中 τ 满足对所有 i, $\tau_{i+1} > \tau_i$ 且对任意 $t \in R$ 存在 i, $\tau_i > t$ 。有时候因建模需要， $\tau_{i+1} > \tau_i$ 可以弱化成 $\tau_{i+1} \geq \tau_i$ 。

定义 $\tau_0 = 0$ 。时间迁移系统 $\langle \Sigma, S, X, \Delta, I\rangle$ 在时间字符串 $(\sigma, \tau) = [(\sigma_i, \tau_i)]_{i\geq 1} \in (\Sigma \times R)^\omega$ 上的一个运行是状态集 $S \times V$ 上的一个无穷序列 $[(s_i, v_i)]_{i\geq 0}$ 满足

$s_0 \in I$,
对所有 $x \in X$,$v_0(x) = 0$,
对任意 $i \geq 0$ ，存在 $\lambda \subseteq X$, $\varphi \in \Phi(X)$, $(s_i, \sigma_{i+1}, \lambda, \varphi, s_{i+1}) \in \Delta$ 使得 $(v_i + \tau_{i+1} - \tau_i)$ 满足 φ 且 $v_{i+1} = [\lambda \to 0](v_i + \tau_{i+1} - \tau_i)$ 。

运行

$S \times V$ 上的一个无穷序列 r 是时间迁移系统 $\langle \Sigma, S, X, \Delta, I\rangle$ 一个运行当且仅当存在一个时间字符串 (σ, τ) 使得 r 是时间迁移系统在 (σ, τ) 上的一个运行。

可达状态

给定时间迁移系统 $\langle \Sigma, S, X, \Delta, I\rangle$ 。一个状态 s 可达当且仅当存在一个 $\langle \Sigma, S, X, \Delta, I\rangle$ 上的运行 $[(s_i, v_i)]_{i\geq 0}$ 和 $i \geq 0$ 使得 $s = s_i$ 。 时间迁移系统的状态可达性问题的复杂度是 PSPACE 完全的。

§2.4.2 时间 Büchi 自动机 (TA)

Definition 2.20 *一个时间 Büchi 自动机是一个六元组*

$$\langle \Sigma, S, X, \Delta, I, F\rangle$$

其中 $\langle \Sigma, S, X, \Delta, I\rangle$ 构成一个时间迁移系统， $F \subseteq S$ 为自动机的接受状态集。

可接受运行

时间 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, X, \Delta, I, F\rangle$ 。一个时间字符串 (σ, τ) 上的 B 的运行 $r = (s, v)$ 是可接受的，当且仅当 $inf(r) \cap F \neq \emptyset$ ，其中 $inf(r)$ 代表无限多次出现在运行 r 中的状态的集合。

可接受时间字符串

若一个时间字符串 (σ, τ) 上存在 B 的可接受运行，则称 (σ, τ) 是可接受。 $\mathcal{B}$ 上可接受时间字符串的集合为 $\mathcal{B}$ 的语言，记作 $\mathcal{L}(\mathcal{B})$ 。 时间自动机是并和交封闭的，但不是补封闭的。时间自动机的语言非空问题的复杂度是 PSPACE 完全的。

§2.4.3 混成迁移系统

混成迁移系统是时间迁移系统的一种扩充。设 $X = \{x_1, ..., x_n\}$ 为实数变量的集合。记 $\Phi(X)$ 为 X 上谓词的集合。记 $\uplus(X) : X \to (R^n \to R)$ 为偏函数的集合。记 $\dot{X} = \{\dot{x_1}, ..., \dot{x_n}\}$ 。

Definition 2.21 *一个混成迁移系统是一个六元组*

$$\langle \Sigma, S, X, \Delta, I, flow\rangle$$

其中 Σ 为字母表， S 为状态集合， X 为实数变量集合， $\Delta \subseteq S \times \Sigma \times \uplus(X) \times \Phi(X) \times S$ 为迁移关系， $I \in S \times (2^R)^n$ 为系统初始状态的集合， $flow : S \to \Phi(X \cup \dot{X})$ 为每个状态标上一个系统变量变化的约束条件。

系统状态和状态的迁移

记 $V = X \to R$ 为变量赋值的集合。一个系统状态为 $S \times V$ 的一个元素。为方便起见，用 $v \in V$ 表示 $(v(x_1), ..., v(x_n)) \in R^n$ 。

对 $\delta \in R_{\geq 0}$ ，用 $(s, v) \xrightarrow{\delta} (s, v')$ 表示 v' 满足以下条件：存在一个可微函数 $f : [0, \delta] \to R^n$ 和其一阶微分 $\dot{f} : (0, \delta) \to R^n$ 满足 $f(0) = v$, $f(\delta) = v'$ 且

$$\forall \zeta \in (0, \delta). flow(s)[X/f(\zeta)][\dot{X}/\dot{f}(\zeta)] = true.$$

用 $(s, v) \xrightarrow{\sigma} (s', v')$ 表示 (s', v') 满足以下条件：存在 $\lambda \in \uplus(X)$, $\varphi \in \Phi(X)$, $(s, \sigma, \lambda, \varphi, s') \in \Delta$ 且 $\{z_1, ..., z_k\} = dom(\lambda)$ 使得 v 满足 φ 且 $v' = v[z_1/\lambda(z_1)(v)] \cdots [z_k/\lambda(z_k)(v)]$ 。

时间字符串上的运行

一个时间字符串为 $\Sigma \times R$ 上的无穷序列。混成迁移系统 $\langle \Sigma, S, X, \Delta, I, flow \rangle$ 在时间字符串 $(\sigma, \tau) = [(\sigma_i, \tau_i)]_{i \geq 1} \in (\Sigma \times R)^\omega$ 上的一个运行是状态集 $S \times V$ 上的一个无穷序列 $[(s_i, v_i)]_{i \geq 0}$ 满足 $(s_0, v_0) \in I$ 且对任意 $i \geq 0$ ，存在 v_i' 使得 $(s_i, v_i) \xrightarrow{\tau_{i+1} - \tau_i} (s_i, v_i')$ 且 $(s_i, v_i') \xrightarrow{\sigma_{i+1}} (s_{i+1}, v_{i+1})$ 。

运行

$S \times V$ 上的一个无穷序列 r 是混成迁移系统 $\langle \Sigma, S, X, \Delta, I, flow \rangle$ 一个运行当且仅当存在一个时间字符串 (σ, τ) 使得 r 是混成迁移系统在 (σ, τ) 上的一个运行。

§2.5 Petri 网

Petri 网的特点是其可以描述真并发。其基本要素有库所、迁移、连接库所和迁移的有向边。由库所进入迁移的边可以看成是输入条件，由迁移进入库所的边可以看成是迁移的输出。

Definition 2.22 一个 *Petri* 网是一个四元组

$$\langle P, T, F, M_0 \rangle$$

其中 P 为库所集合， T 为迁移集合， $F \subseteq (P \times T) \cup (T \times P)$ 为边的集合， $M_0 : P \to N$ 为初始状态，其中 N 为自然数集合。

系统状态

一个系统状态为 $P \to N$ 的一个函数，是对所有库所的一个赋值。

迁移

定义 $^\circ p(t) = \{p \in P | (p, t) \in F\}$ 和 $p^\circ(t) = \{p \in P | (t, p) \in F\}$ 。迁移 $t \in T$ 在状态 M 是可执行的，当且仅当输入条件满足，即 $\forall p \in^\circ p(t), M(p) \geq 1$ 。 用 $M \xrightarrow{t} M'$ 表示 $t \in T$ 在状态 M 是可执行且 $\forall p \in P$ ， $M'(p) = M(p) - \alpha_0(p, t) + \alpha_1(p, t)$ 其中

$\alpha_0(p, t) = 1$ 当且仅当 $p \in^\circ p(t)$
$\alpha_1(p, t) = 1$ 当且仅当 $p \in p^\circ(t)$

可达状态

给定 Petri 网 $Q = \langle P, T, F, M_0 \rangle$ 。用 $M \rightarrow M'$ 表示存在 $t \in T$ 使得 $M \xrightarrow{t} M'$ 成立。则 Q 的可达状态集合可表示为 $\{M | M_0 \xrightarrow{*} M\}$ 。 Petri 网的状态可达性问题的复杂度是可判定的且是 EXPSPACE 难的。

安全 Petri 网

Petri 网 $\langle P, T, F, M_0 \rangle$ 是 k 界的，当且仅当对其所有可达状态 M 有 $\forall p \in P.M(p) \leq k$ 。 1 界 Petri 网称为安全 Petri 网。安全 Petri 网的状态可达性问题的复杂度是 PSPACE 完备的。

§2.6 通信系统

通信系统模型用来表达多个进程通过通道交换信息进行控制和计算的过程。每个进程内部有状态和状态迁移。状态迁移的原因可以是内部事件或输入输出。

§2.6.1 通道

Definition 2.23 给定一个类型 $\langle S, N \rangle$ 其中 S 为字母表的， N 为自然数。一个类型为 $\langle S, N \rangle$ 的通道 m ，记作 $m \in \langle S, N \rangle$ ，是一个值域为 $\bigcup_{i=0}^{N} \{\langle x_1, ..., x_i \rangle | x_i \in S\}$ 的变量。

通道状态记载通道的给定赋值，记为 σ 。若 $m \in \langle S, N \rangle$ ，则 $\sigma(m) \in \bigcup_{i=0}^{N} \{\langle x_1, ..., x_i \rangle | x_i \in S\}$ 。通道状态空间为通道状态的集合，记为 Σ 。

事件

若 $m \in \langle S, N \rangle$, 则与 m 相关的事件集合 $\alpha(m)$ 定义为 $\alpha(m) = \{m?s | s \in S\} \cup \{m!s | s \in S\}$ 。设 $C = \{m_1, ..., m_n\}$ 为通道集合。则与 C 相关的事件集合为 $\alpha(C) = \alpha(m_1) \cup \cdots \cup \alpha(m_n)$ 。与通道无关的内部事件记为 $\{\epsilon\}$ 。

可执行事件

对于 $x = \langle x_1, ..., x_n \rangle$ ，定义 $|x| = n$, $x \vdash s = \langle x_1, ..., x_n, s \rangle$, $HEAD(x) = x_1$, $TAIL(x) = \langle x_2, ..., x_n \rangle$ 。给定一个通道状态 σ 和一个通道 $m \in \langle S, N \rangle$ 。若以下一项成立则 $a \in \alpha(m) \cup \{\epsilon\}$ 可执行。

$a = \epsilon$
$a = m!s$ 且 $\lvert\sigma(m)\rvert < N$ 且 $s \in S$
$a = m?s$ 且 $\lvert\sigma(m)\rvert > 0$ 且 $s = HEAD(\sigma(m))$

给定 $\sigma \in \Sigma$,$m \in \langle S, N \rangle$,$a \in \alpha(m)$ 。用 $\sigma \xrightarrow{a} \sigma'$ 表示 a 在 σ 可执行且

若 $a = \epsilon$,	则 $\sigma' = \sigma$
若 $a = m!s$,	则 $\sigma' = \sigma[m/\sigma(m) \vdash s]$
若 $a = m?s$,	则 $\sigma' = \sigma[m/TAIL(\sigma(m))]$

§2.6.2 通信单元

Definition 2.24 一个通信单元是一个四元组

$$\langle Q, C, \Delta, q_0 \rangle$$

其中 Q 为状态集合， C 为通道集合， $\Delta \subseteq Q \times (\alpha(C) \cup \{\epsilon\}) \times Q$ 为标号的迁移关系， $q_0 \in Q$ 为初始状态。

系统状态

给定一个通信单元 $\langle Q, C, \Delta, q_0 \rangle$ ，系统状态由两部分组成 (s, σ) 其中 $s \in Q$ 为控制状态，$\sigma \in \Sigma$ 为通道状态。

运行

迁移 (q, a, q') 在 (s, σ) 可执行，当且仅当 $q = s$ 且 a 在 σ 可执行。 设 $C = \{m_1, ..., m_k\}$ 。一个运行是 $Q \times \Sigma$ 上的一个无穷序列 $[(z_i, \sigma_i)]_{i \geq 0}$ 满足 $z_0 = q_0$, $\sigma_0(m_1) = \cdots = \sigma_0(m_k) = \langle\rangle$ 且对任意 $i \geq 0$,

存在 $(q, a, q') \in \Delta$
(q, a, q') 在 (z_i, σ_i) 可执行
$z_{i+1} = q'$ 且 $\sigma_i \xrightarrow{a} \sigma_{i+1}$

§2.6.3 通信系统

Definition 2.25 一个通信系统 $\{P_1, ..., P_n\}$ 是由通信单元 $P_1 = \langle Q_1, C_1, \Delta_1, q_{10} \rangle$, ..., $P_n = \langle Q_n, C_n, \Delta_n, q_{n0} \rangle$ 组成的集合，其中 $Q_1, ..., Q_n$ 为两两不相交的集合。

系统状态

定义 $S_1 \otimes \cdots \otimes S_n = \{\{s_1, ..., s_n\} | s_1 \in S_1, ..., s_n \in S_n\}$ 。通信系统的系统状态的集合为

$$(Q_1 \otimes \cdots \otimes Q_n) \times \Sigma$$

运行

一个迁移 $(q, a, q') \in \Delta_1 \cup \cdots \cup \Delta_n$ 在状态 (Q, σ) 可执行当且仅当 $q \in Q$ 且 a 在 σ 可执行。设 $C_1 \cup \cdots \cup C_n = \{m_1, ..., m_k\}$ 。一个运行是 $(Q_1 \otimes \cdots \otimes Q_n) \times \Sigma$ 上的一个无穷序列 $[(z_i, \sigma_i)]_{i \geq 0}$ 满足 $z_0 = \{q_0, ..., q_n\}$, $\sigma_0(m_1) = \cdots = \sigma_0(m_k) = \langle\rangle$ 且对任意 $i \geq 0$,

存在 $(q, a, q') \in \Delta_1 \cup \cdots \cup \Delta_n$
(q, a, q') 在 (z_i, σ_i) 可执行
$z_{i+1} = (z_i \setminus \{q\}) \cup \{q'\}$ 且 $\sigma_i \xrightarrow{a} \sigma_{i+1}$

并发算子

给定两个通信单元 $P_1 = \langle Q_1, C_1, \Delta_1, q_{10}\rangle$, $P_2 = \langle Q_2, C_2, \Delta_2, q_{20}\rangle$ 且 $Q_1 \cap Q_2 = \emptyset$ 。 P_1 和 P_2 的并发定义如下

$$P_1||P_2 = \langle Q_1 \otimes Q_2, C_1 \cup C_2, \Delta, \{q_{10}, q_{20}\}\rangle$$

其中

$$\Delta = \{(\{q_1, q_2\}, a, \{q_1', q_2\})|(q_1, a, q_1') \in \Delta_1\} \cup \{(\{q_1, q_2\}, a, \{q_1, q_2'\})|(q_2, a, q_2') \in \Delta_2\}$$

并发算子满足结合率和交换率。

§2.7 说明

本章主要内容为程序及系统运行的模型。有关卫式迁移系统部分可参考 [Pel01] 。 Kripke 模型和自动机可参考 [CGP99] 。交错自动机部分可参考 [Var97, AHK97] 。时间自动机部分可参考 [AD94] 。混成自动机部分可参考 [Hen96] 。 Petri 网的介绍可参考 [NW96] 。 通信系统部分可参考 [Hol90] 。

参考文献

[AD94] Rajeev Alur and David L. Dill. A Theory of Timed Automata. Theor. Comput. Sci. 126(2): 183-235 (1994).

[AHK97] Rajeev Alur, Thomas A. Henzinger and Orna Kupferman. Alternating-Time Temporal Logic. COMPOS 1997: 23-60.

[CGP99] E. Clark, O. Grumberg and D. Peled. Model Checking. MIT press, 1999.

[Hen96] Thomas A. Henzinger. The Theory of Hybrid Automata. LICS 1996: 278-292.

[Hol90] Gerard J. Holzmann. Design and Validation of Computer Protocols. Prentice Hall, 1990.

[NW96] Mogens Nielsen and Glynn Winskel. Petri Nets and Bisimulation. Theor. Comput. Sci. 153(1&2): 211-244 (1996).

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

[Var97] Moshe Y. Vardi. Alternating Automata: Unifying Truth and Validity Checking for Temporal Logics. CADE 1997: 191-206.

中国科学院研究生院课程讲稿

§3 程序逻辑

为了能够表达程序性质，进而验证程序是否具有这样的性质，我们需要描述程序性质的语言，即程序逻辑。 程序逻辑的基础是命题逻辑、一阶逻辑和模态逻辑。命题逻辑和一阶逻辑作为程序的断言是表示程序性质的一种手段。 在命题逻辑和一阶逻辑的基础之上，我们引进模态算子并将其含义解释成与时间先后次序有关的描述，形成不同的时序逻辑。本章介绍时序逻辑。

§3.1 线性时序逻辑

线性时序逻辑 关心的是系统运行中的状态以及它们之间的关系。线性时序逻辑可以建立在命题逻辑、谓词逻辑以及其它描述状态的逻辑之上。

§3.1.1 命题线性时序逻辑 (PLTL)

我们先考虑建立在命题逻辑上的线性时序逻辑，即命题线性时序逻辑，记作 PLTL 。 给定一个原子命题集合 AP 。 PLTL 公式的集合由以下语法给出。

$$\phi ::= \quad \bot \mid \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \to \phi \mid \phi \leftrightarrow \phi \mid O\,\phi \mid \Diamond\,\phi \mid \Box\,\phi \mid \phi\, U\, \phi \mid \phi\, R\, \phi$$

其中 p 为 AP 中任意命题。

解释 1

AP 上的 PLTL 公式在一个字母表为 2^{AP} 的字符串上解释。 设 φ 是 PLTL 公式。设 $\zeta = \zeta_0\zeta_1 \cdots \in (2^{AP})^\omega$ 。我们用 ζ^k 来表示 ζ_k 开始的状态序列 $\zeta_k\zeta_{k+1}\cdots$ 。则 PLTL 公式的语义如下：

$\zeta \models p$	若 $p = \top$, 或 $p \in AP$ 且 $p \in \zeta_0$
$\zeta \models \neg\varphi$	若 $\zeta \not\models \varphi$
$\zeta \models \varphi \vee \psi$	若 $\zeta \models \varphi$ 或 $\zeta \models \psi$
$\zeta \models \varphi \wedge \psi$	若 $\zeta \models \varphi$ 且 $\zeta \models \psi$
$\zeta \models \varphi \to \psi$	若 $\zeta \models \varphi$ 则 $\zeta \models \psi$
$\zeta \models \varphi \leftrightarrow \psi$	若 $\zeta \models \varphi \to \psi$ 且 $\zeta \models \psi \to \varphi$
$\zeta \models O\varphi$	若 $\zeta^1 \models \varphi$
$\zeta \models \Box\psi$	若 $\forall i \geq 0, \zeta^i \models \psi$
$\zeta \models \Diamond\varphi$	若 $\exists i \geq 0, \zeta^i \models \varphi$
$\zeta \models \varphi U \psi$	若 $\exists i \geq 0, \zeta^i \models \psi$ 且 $\forall 0 \leq j < i, \zeta^j \models \varphi$
$\zeta \models \varphi R \psi$	若 $\forall i \geq 0$, 若 $(\forall 0 \leq j < i, \zeta^j \not\models \varphi)$ 则 $\zeta^i \models \psi$

解释 2

给定标号 Kripke 结构 $M = \langle S, R, I, L\rangle$ 。设 $\pi = \pi_0\pi_1\pi_2 \cdots \in (S)^\omega$ 为一个无限状态序列。定义 $L(\pi) = L(\pi_1)L(\pi_2)\cdots$ 。

$$M, \pi \models \varphi \quad \text{当且仅当} \quad L(\pi) \models \varphi \text{ 。}$$

设 φ 为 PLTL 公式， M 为标号 Kripke 结构。

$$M \models \varphi$$

当且仅当

$$\text{对所有 } M \text{ 的运行 } \pi \text{ ，有 } M, \pi \models \varphi \text{ 。}$$

G, F, R 和 U 的递推关系

关于 R 和 U ，我们有以下等式。

Gq	$\equiv$	$(q \wedge O(Gq))$
Fq	$\equiv$	$(q \vee O(Fq))$
pRq	$\equiv$	$(q \wedge (p \vee O(pRq)))$
pUq	$\equiv$	$(q \vee (p \wedge O(pUq)))$

模态算子的对偶关系与 NNF 范式

只使用逻辑连接符 $\wedge, \vee, \neg$ 且逻辑连接符 $\neg$ 只出现在命题前面的公式称为 NNF 范式。每个 PLTL 公式等价于一个 PLTL 的 NNF 范式。 一个 PLTL 的 NNF 范式可应用以下对偶关系构造。

$(\Box p)$	$\equiv$	$\neg(\Diamond \neg p)$
(pRq)	$\equiv$	$\neg(\neg pU \neg q)$
(Op)	$\equiv$	$\neg(O \neg p)$

模态算子的极小完全集

除了以上对偶的算子之外，我们有以下等式。

$\Box p$	$\equiv$	$(\bot R p)$
$\Diamond p$	$\equiv$	$(\top U p)$

因此 $\{O, U\}$ 构成 PLTL 模态算子的一个完全集，且是极小完全集。

§3.1.2 PLTL 公式的推理

以 $\{O, U, \Box, \Diamond\}$ 为 PLTL 公式的模态算子集。 PLTL 的推理系统包含以下三部分：一部分为 PLTL 公式相关的时序逻辑公理；另一部分为命题逻辑的推理系统；第三部分为时序推理规则。

第一部分：公理

$\Box \neg p \leftrightarrow \neg \Diamond p$
$\Box(p \to q) \to (\Box p \to \Box q)$
$\Box p \to p$
$\Box p \to Op$
$\Box p \to O \Box p$
$\Box(p \to Op) \to (p \to \Box p)$
$O \neg p \leftrightarrow \neg Op$
$O(p \to q) \to (Op \to Oq)$
$(pUq) \leftrightarrow (q \vee (p \wedge O(pUq))$
$(pUq) \to \Diamond q$

第二部分：命题逻辑推理系统

公理： 如果 p 是重言式，则 p 是公理。
MP 规则： 如果 $\vdash p \to q$ 且 $\vdash p$ ，则 $\vdash q$ 。

第三部分：时序推理规则

如果 $\vdash p$ ，则 $\vdash \Box p$（时序推广）。

可满足性

PLTL 公式的可满足性是可判定的，其判定复杂性为 PSPACE 完全。 只允许时序算子 $\Box$ 和 $\Diamond$ 的 PLTL 公式的子集记为 PLTL(F) 。 PLTL(F) 的判定复杂性为 NP 完全。

§3.1.3 PLTL 公式的不动点表示

给定 $\zeta = \zeta_0\zeta_1 \cdots \in (2^{AP})^\omega$ 和 PLTL 公式 φ 。将 $\mathbf{N}$ 的子集 $\{i \mid \zeta^k \models \varphi\}$ 记为 $[[\varphi]]$ 。
定义函数 $o : 2^{\mathbf{N}} \to 2^{\mathbf{N}}$ 如下：

$$o(A) = \{i \in \mathbf{N} \mid i+1 \in A\}$$

我们有以下等式：

$[[p]]$	$=$	$\{i \mid p \in \zeta_i\}$
$[[\neg\varphi]]$	$=$	$\mathbf{N} \setminus [[\varphi]]$
$[[\varphi \wedge \psi]]$	$=$	$[[\varphi]] \cap [[\psi]]$
$[[O\varphi]]$	$=$	$\{i \mid i+1 \in [[\varphi]]\}$
$[[\varphi U \psi]]$	$=$	$[[\psi]] \cup ([[\varphi]] \cap [[O(\varphi U \psi)]]$

由以上等式知 $[[\varphi U\psi]]$ 是 $\tau(Z) = [[\psi]] \cup ([[\varphi]] \cap ex(Z))$ 的不动点。 根据 PLTL 语义知 τ 有最小和最大不动点，且 $[[\varphi U\psi]]$ 是 τ 的最小不动点，即

$$[[[[\varphi U\psi]]]] \quad = \quad \mu Z([[\psi]] \cup ([[\varphi]] \cap ex(Z)))$$

类似地，其它模态算子也可以用不动点刻画。为方便书写，我们直接将 p, q 看成 $\mathbf{N}$ 的子集，模态算子和逻辑联接符看成是 $2^{\mathbf{N}}$ 上的函数。

Fp	$=$	$\mu Z(p \vee OZ)$
Gp	$=$	$\nu Z(p \wedge OZ)$
pUq	$=$	$\mu Z(q \vee (p \wedge OZ))$
pRq	$=$	$\nu Z(q \wedge (p \vee OZ))$

给定 $\zeta = \zeta_0\zeta_1 \cdots \in (2^{AP})^\omega$ 和 PLTL 公式 φ 。我们有

$$\zeta \models \varphi \quad \text{当且仅当} \quad 0 \in [[\varphi]] \text{ 。}$$

§3.1.4 PLTL 公式与 Büchi 自动机

记 $[[\varphi]] \subseteq (2^{AP})^\omega$ 为满足 φ 的字符串的集合。对任意 PLTL 公式 φ , 存在一个 Büchi 自动机 $A = \langle 2^{AP}, S, \Delta, I, F\rangle$ 使得 $\mathcal{L}(A) = [[\varphi]]$ 。 由于每个 PLTL 公式等价于一个 NNF 范式且 $\Box$ 和 $\Diamond$ 可用 R 和 U 表示, 我们只考虑不含 $\Box$ 和 $\Diamond$ 的 NNF 范式。由于 Büchi 自动机可通过扩展 Büchi 自动机的转换得到，以下是一个构造语言等价于 $[[\varphi]]$ 的扩展 Büchi 自动机 $B_\varphi = \langle 2^{AP}, S, \Delta, I, F\rangle$ 的方法。

1. 首先构造状态 $x = (\{\varphi\}, \{\}, \{\})$，记 $a(x) = \{\epsilon\}$。将其加入状态集 S。

2. 对任意 $x = (b, c, d) \in S$：

若 $b = \emptyset$，构造状态 $y = (d, \{\}, \{\})$。若 $y \in S$，则修改 $a(y) = a(y) \cup \{x\}$，若 $y \notin S$，记 $a(y) = \{x\}$，并将 y 加入状态集 S。

若 $b = b_0 \cup \{\varphi\}$，我们分以下几种情况：

若 $\varphi = \varphi_0 U \varphi_1$，构造状态 $(b_0 \cup \{\varphi_0\}, c \cup \{\varphi\}, d \cup \{\varphi\})$ 和 $(b_0 \cup \{\varphi_1\}, c \cup \{\varphi\}, d)$。

若 $\varphi = \varphi_0 R \varphi_1$，构造状态 $(b_0 \cup \{\varphi_1\}, c \cup \{\varphi\}, d \cup \{\varphi\})$ 和 $(b_0 \cup \{\varphi_0, \varphi_1\}, c \cup \{\varphi\}, d)$。

若 $\varphi = \varphi_0 \vee \varphi_1$，构造状态 $(b_0 \cup \{\varphi_0\}, c \cup \{\varphi\}, d)$ 和 $(b_0 \cup \{\varphi_1\}, c \cup \{\varphi\}, d)$。

若 $\varphi = \varphi_0 \wedge \varphi_1$，构造状态 $(b_0 \cup \{\varphi_0, \varphi_1\}, c \cup \{\varphi\}, d)$。

若 $\varphi = O\varphi_0$，构造状态 $(b_0, c \cup \{\varphi\}, d \cup \{\varphi_0\})$。

若 $\varphi = p$ 或 $\varphi = \neg p$，且 $p \in AP$，构造状态 $(b_0, c \cup \{\varphi\}, d)$。

用新构造的状态替代原状态 x。首先对每个新构造的状态 y，若 $y \notin S$，记 $a(y) = a(x)$，并将 y 加入状态集 S；若 $y \in S$ 且 $a(y) \neq \emptyset$，则修改 $a(y) = a(y) \cup a(x)$; 若 $y \in S$ 且 $a(y) = \emptyset$，则对所有直接或间接替代 y 的状态 $y' \in S$，修改 $a(y') = a(y') \cup a(x)$; 然后将所有 $a(z)$ 中出现 x 的地方用新构造的状态替代；最后修改 $a(x) = \emptyset$ 表示 x 已为新构造的状态所替代。

对每个新构造的状态 $y = (b, c, d) \in S$，若 c 中命题的集合是不可满足的，则从 S 中删除 y；若 $c \cap b \neq \emptyset$，则从 b 中删除公共部分 $c \cap b$。返回第二步直至没有新构造的状态。

3. 对任意 $x = (b, c, d) \in S$，若 $y \in a(x)$ 且 $y = (b', c', d') \in S$，构造迁移 (y, σ, x)，并将其加入迁移集合 Δ，其中 $\sigma \in 2^{AP}$ 满足 $\forall p \in AP.(\{p, \neg p\} \not\subseteq \sigma \cup c') \wedge (p \in c' \to p \in \sigma)$。

4. 对任意 $x = (b, c, d) \in S$，若 $\epsilon \in a(x)$，则将 x 加入初始状态集合 I。

5. 对每个公式 φ，若 $\varphi = \varphi_0 U \varphi_1$，构造 $f_\varphi = \{(b, c, d) \in S \mid \varphi \in c \to \varphi_1 \in c\}$，并将其加入接受状态集集合 F。

§3.1.5 一阶线性时序逻辑

一阶线性时序逻辑，即用一阶逻辑来描述状态性质，不是可判定的，并且不存在完备的一阶线性时序逻辑的推理系统。但其表达能力较强，能够表达更多的性质，可以用在适合推理的程序验证中。 给定一个一阶逻辑 $\mathcal{L}^B$， $\mathcal{L}^B$ 上的一阶线性时序逻辑公式的集合由以下语法给出。

$$\phi ::= \quad p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \to \phi \mid \phi \leftrightarrow \phi \mid O\,\phi \mid \Diamond\,\phi \mid \Box\,\phi \mid \phi\, U\, \phi \mid \phi\, R\, \phi$$

其中 p 为 $\mathcal{L}^B$ 的任意公式。 一阶线性时序逻辑公式在无限状态序列上解释。设 S 为状态集合， $\pi = \pi_0\pi_1\pi_2\cdots$ 为一个无限状态序列， S 中状态的结构及在其之上成立的公式决定于一阶逻辑 $\mathcal{L}^B$。一阶线性时序逻辑公式的语义如下：

$\pi^k \models p$	若 p 是 $\mathcal{L}_B$ 的公式且 $\pi_k \models p$
$\pi^k \models \neg\varphi$	若 $\pi^k \not\models \varphi$
$\pi^k \models \varphi \vee \psi$	若 $\pi^k \models \varphi$ 或 $\pi^k \models \psi$
$\pi^k \models \varphi \wedge \psi$	若 $\pi^k \models \varphi$ 且 $\pi^k \models \psi$
$\pi^k \models \varphi \to \psi$	若 $\pi^k \models \varphi$ 则 $\pi^k \models \psi$
$\pi^k \models \varphi \leftrightarrow \psi$	若 $\pi^k \models \varphi \to \psi$ 且 $\pi^k \models \psi \to \varphi$
$\pi^k \models O\varphi$	若 $\pi^{k+1} \models \varphi$
$\pi^k \models \Box\psi$	若 $\forall i \geq k, \pi^i \models \psi$
$\pi^k \models \Diamond\varphi$	若 $\exists i \geq k, \pi^i \models \varphi$
$\pi^k \models \varphi U \psi$	若 $\exists i \geq k, \pi^i \models \psi$ 且 $\forall k \leq j < i, \pi^j \models \varphi$
$\pi^k \models \varphi R \psi$	若 $\forall i \geq k$, 若 $\forall k \leq j < i, \pi^j \not\models \varphi$ 则 $\pi^i \models \psi$

推理规则

我们用 $\varphi \Rightarrow \psi$ 表示 $\Box(\varphi \to \psi)$ 。给定一个 $B = (F, P)$ 和 B 的解释 I 。根据一阶时序逻辑公式的语义，我们有以下推理规则。

- 推广规则：

$$\frac{\vdash \varphi}{\Box\varphi} \qquad \frac{\varphi \Rightarrow \psi}{\varphi \Rightarrow \varphi U \psi} \qquad \frac{\varphi \Rightarrow O\psi}{\varphi \Rightarrow \varphi U \psi}$$

- R 规则：

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

- 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

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

一阶线性时序逻辑的扩展

以上一阶线性时序逻辑的模态算子都是作用在公式上。更进一步，我们可以允许 O 算子用在项上。这样，我们需要修改 $M, \pi^k \models p$ 的定义。设 p 是 $\mathcal{L}^B$ 的带有 n 个变量的原子公式，则：

$M, \pi^k \models p(O^{i_1}x_1, ..., O^{i_n}x_n)$	若 O 不在 p 中的其它位置出现 且 $M, \pi^k \models p(\pi_{k+i_1}(x_1), ..., \pi_{k+i_n}(x_n))$
$M, \pi^k \models p(..., O^i f(t_1, ..., t_m), ...)$	若 $M, \pi^k \models p(..., f(O^i t_1, ..., O^i t_m), ...)$

§3.1.6 线性 μ- 演算 (νTL)

由关于 PLTL 的讨论我们知道 PLTL 公式可以用模态算子 O, 命题变量和不动点表示。在这种表示中，命题变量可以出现在 O, μ, ν 之后。将这个约束去掉，我们可以得到一个表达能力更强的逻辑。称为 νTL 。 给定一个原子命题集合 AP ，一个变量集合 V 。 νTL 公式的集合由以下语法给出。

$$\phi ::= p \mid X \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid O\phi \mid \mu X.\phi \mid \nu X.\phi$$

其中 p 为 AP 中任意命题， X 为变量，并且 $\mu X.\phi$ 和 $\nu X.\phi$ 的 ϕ 中不受囿的 X 必须在偶数个 $\neg$ 符号的作用范围之下。

νTL 公式在线性结构上的语义解释

νTL 公式可在线性结构上解释。则 νTL 公式有以下语义。设 $M = (\pi, L)$ 为一线性结构，其中 $\pi = \pi_0\pi_1\cdots$ 为状态序列， $L : \{\pi_0, \pi_1, ...\} \to 2^{AP}$ 为状态集到命题幂集的映射。设 $e : V \to 2^N$ 为变量到 N 子集的赋值。 νTL 公式的语义如下：

$$\begin{array}{lcl}
[[p]]e & = & \{i \in N \mid p \in L(\pi_i)\} \\
[[X]]e & = & e(X) \\
[[\neg\phi]]e & = & N \setminus [[\phi]]e \\
[[\phi_1 \wedge \phi_2]]e & = & [[\phi_1]]e \cap [[\phi_2]]e \\
[[\phi_1 \vee \phi_2]]e & = & [[\phi_1]]e \cup [[\phi_2]]e \\
[[O\phi]]e & = & \{i \in N \mid i+1 \in [[\phi]]e\} \\
[[\mu X.\phi]]e & = & \cap\{S' \subseteq N \mid [[\phi]]e[S'/X] \subseteq S'\} \\
[[\nu X.\phi]]e & = & \cup\{S' \subseteq N \mid S' \subseteq [[\phi]]e[S'/X]\}
\end{array}$$

$M \models \varphi$ 当且仅当 $0 \in [[\varphi]]$ 。

模态算子的对偶关系与 NNF 范式

只使用逻辑连接符 $\wedge, \vee, \neg$ 且逻辑连接符 $\neg$ 只出现在命题前面的公式称为 NNF 范式。每个 νTL 公式等价于一个 νTL 的 NNF 范式。 一个 νTL 的 NNF 范式可应用以下对偶关系构造。

$$\begin{array}{lcl}
\mu X.\phi & \equiv & \neg(\nu X.\neg\phi[\neg X/X]) \\
(Op) & \equiv & \neg(O\neg p)
\end{array}$$

一个公式是受佑的，当且仅当所有变量都受其不动点算子下的一个 O 算子的约束。所有闭公式等价于一个 NNF 受佑公式。

推理系统

根据 νTL 公式的语义，我们有以下推理规则。

$$\frac{\vdash \Gamma, \varphi, \psi}{\vdash \Gamma, \varphi \vee \psi} \qquad \frac{\vdash \Gamma, \varphi \qquad \vdash \Gamma, \psi}{\vdash \Gamma, \varphi \wedge \psi} \qquad \frac{\vdash \Gamma, \varphi[\sigma X.\varphi/X]}{\vdash \Gamma, \sigma X.\varphi} \qquad \frac{\vdash \Gamma}{\vdash O\Gamma, \Delta}$$

以上推理系统对 NNF 受佑公式是可靠完备的。

νTL 公式在 KS 上的语义解释

νTL 公式亦可在 AP 上的 KS 上解释。设 R 是 KS 上以任意状态为起点的所有运行的集合。设 M 为 AP 上的 KS， $e : V \to 2^R$ 为变量到 R 子集的赋值。 νTL 公式的语义如下：

$$\begin{array}{lcl}
[[p]]e & = & \{\pi \mid p \in L(\pi_0)\} \\
[[X]]e & = & e(X) \\
[[\neg\phi]]e & = & R \setminus [[\phi]]e \\
[[\phi_1 \wedge \phi_2]]e & = & [[\phi_1]]e \cap [[\phi_2]]e \\
[[\phi_1 \vee \phi_2]]e & = & [[\phi_1]]e \cup [[\phi_2]]e \\
[[O\phi]]e & = & \{\pi \mid \pi_1 \in [[\phi]]e\} \\
[[\mu X.\phi]]e & = & \cap\{R' \subseteq R \mid [[\phi]]e[R'/X] \subseteq R'\} \\
[[\nu X.\phi]]e & = & \cup\{R' \subseteq R \mid R' \subseteq [[\phi]]e[R'/X]\}
\end{array}$$

设 M 为 KS， φ 为闭公式。φ 在 M 中的语义可写为 $[[\varphi]]$ 。$M, \pi \models \varphi$ 当且仅当 $\pi \in [[\varphi]]$ 。记 M 的运行集合为 $[[M]]$ 。

$$M \models \varphi$$

当且仅当

$$[[M]] \subseteq [[\varphi]] \text{ 。}$$

LTL 公式到 νTL 公式的转换

LTL 公式可以看成是 νTL 的一个子类，可以用以下规则将 LTL 公式转换到 νTL 公式。

$$\begin{array}{lcl}
T(p) & = & p \\
T(\neg\varphi) & = & \neg T(\varphi) \\
T(\varphi \wedge \psi) & = & T(\varphi) \wedge T(\psi) \\
T(O\varphi) & = & OT(\varphi) \\
T(\varphi U \psi) & = & \mu Y.(T(\psi) \vee (T(\varphi) \wedge OY))
\end{array}$$

§3.2 分枝时序逻辑

系统的可能的运行可以看成是一个树的结构，即一个状态可能有多个后续状态。一个系统由一颗或多颗树组成。分枝时序逻辑可以描述计算树的分枝情况和状态的前后关系。 分枝时序逻辑可以建立在命题逻辑、谓词逻辑以及其它描述状态的逻辑之上。

§3.2.1 计算树逻辑 CTL 的语法和语义

我们考虑一种简单的计算树逻辑，即 CTL 。 CTL 中描述分枝情况和描述状态的前后关系的算子成对出现，即一个描述分枝情况的算子后面必须有一个描述状态的前后关系的算子。 我们考虑命题逻辑之上的计算树逻辑。给定一个原子命题集合 AP 。 CTL 公式的集合由以下语法给出。

$$\begin{array}{rl}
\phi ::= & \top \mid \bot \mid p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \\
& EX\ \phi \mid EF\ \phi \mid EG\ \phi \mid E(\phi\, U\, \phi) \mid E(\phi\, R\, \phi) \mid \\
& AX\ \phi \mid AF\ \phi \mid AG\ \phi \mid A(\phi\, U\, \phi) \mid A(\phi\, R\, \phi)
\end{array}$$

其中 p 为 AP 中任意命题。为间单起见，逻辑联接符 $\to$ 和 $\leftrightarrow$ 没有出现在上面的公式定义中，但我们可以根据其通常的语义自由使用。 CTL 公式在一个树状结构或 Kripke 结构上解释。设 $M = \langle S, \Delta, I, L\rangle$ 是原子命题集 AP 上的 Kripke 结构。 CTL 公式的语义如下：

$M, s \models p$	若 $p = \top$, 或 $p \in AP$ 且 $p \in L(s)$
$M, s \models \neg\varphi$	若 $M, s \not\models \varphi$
$M, s \models \varphi \vee \psi$	若 $M, s \models \varphi$ 或 $M, s \models \psi$
$M, s \models \varphi \wedge \psi$	若 $M, s \models \varphi$ 且 $M, s \models \psi$
$M, s \models A\varphi$	若对于所有 M 中以 s 为起点的路径 π： $M, \pi \models \varphi$
$M, s \models E\varphi$	若存在 M 中以 s 为起点的路径 π： $M, \pi \models \varphi$
$M, \pi \models X\varphi$	若 $M, \pi_1 \models \varphi$
$M, \pi \models G\psi$	若 $\forall i \geq 0, M, \pi_i \models \psi$
$M, \pi \models F\varphi$	若 $\exists i \geq 0, M, \pi_i \models \varphi$
$M, \pi \models \varphi U \psi$	若 $\exists i \geq 0, M, \pi_i \models \psi$ 且 $\forall 0 \leq j < i, M, \pi_j \models \varphi$
$M, \pi \models \varphi R \psi$	若 $\forall i \geq 0$, 若 $\forall 0 \leq j < i, M, \pi_j \not\models \varphi$ 则 $M, \pi_i \models \psi$

设 φ 为 CTL 公式， M 为 Kripke 结构。

$$M \models \varphi$$

当且仅当

$$\text{对所有 } s \in I\text{， } M, s \models \varphi \text{。}$$

G, F, R 和 U 的递推关系

关于 R 和 U ，我们有以下等式。

EGq	$\equiv$	$(q \wedge EXEGq))$
EFq	$\equiv$	$(q \vee EXEFq))$
$E(pRq)$	$\equiv$	$(q \wedge (p \vee EXE(pRq)))$
$E(pUq)$	$\equiv$	$(q \vee (p \wedge EXE(pUq)))$
AGq	$\equiv$	$(q \wedge AXAGq))$
AFq	$\equiv$	$(q \vee AXAFq))$
$A(pRq)$	$\equiv$	$(q \wedge (p \vee AXA(pRq)))$
$A(pUq)$	$\equiv$	$(q \vee (p \wedge AXA(pUq)))$

模态算子的对偶关系与 NNF 范式

逻辑连接符 $\neg$ 只出现在命题前面的公式称为 NNF 范式。每个 CTL 公式等价于一个 CTL 的 NNF 范式。一个 CTL 的 NNF 范式可应用以下对偶关系构造。

AXp	$\equiv$	$\neg EX\neg p$
AGp	$\equiv$	$\neg EF\neg p$
AFp	$\equiv$	$\neg EG\neg p$
$A(pRq)$	$\equiv$	$\neg E(\neg pU\neg q)$
$A(pUq)$	$\equiv$	$\neg E(\neg pR\neg q)$

模态算子的极小完全集

除了以上对偶的算子之外，我们有以下等式。

EFp	$\equiv$	$E(\top Up)$
$E(pRq)$	$\equiv$	$E(qU(p \wedge q)) \vee EGq$

因此 $\{EX, EG, EU\}$ 构成 CTL 模态算子的一个完全集，且是极小完全集。

§3.2.2 CTL 公式的推理

以 $\{EX, EU, EG, EF, AX, AU, AG, AF, \}$ 为 CTL 公式的模态算子集。 CTL 的推理系统包含以下三部分：一部分为 CTL 式相关的时序逻辑公理；另一部分为命题逻辑的推理系统；第三部分为时序推理规则。

第一部分：公理

$EFp \leftrightarrow E(\top U p)$
$AGp \leftrightarrow \neg EF\neg p$
$AFp \leftrightarrow A(\top U p)$
$EGp \leftrightarrow \neg AF\neg p$
$EX(p \vee q) \leftrightarrow EXp \vee EXq$
$AXp \leftrightarrow \neg EX\neg p$
$E(pUq) \leftrightarrow (q \vee (p \wedge EXE(pUq))$
$A(pUq) \leftrightarrow (q \vee (p \wedge AXA(pUq))$
$EX\top \wedge AX\top$
$AG(r \rightarrow (\neg q \wedge EXr)) \rightarrow (r \rightarrow \neg A(pUq))$
$AG(r \rightarrow (\neg q \wedge EXr)) \rightarrow (r \rightarrow \neg AFq)$
$AG(r \rightarrow (\neg q \wedge (p \rightarrow AXr))) \rightarrow (r \rightarrow \neg E(pUq))$
$AG(r \rightarrow (\neg q \wedge AXr)) \rightarrow (r \rightarrow \neg EFq)$
$AG(p \rightarrow q) \rightarrow (EXp \rightarrow EXq)$

第二部分：命题逻辑推理系统

公理： 如果 p 是重言式，则 p 是公理。

MP 规则： 如果 $\vdash p \rightarrow q$ 且 $\vdash p$ ，则 $\vdash q$ 。

第三部分：时序推理规则

如果 $\vdash p$ ，则 $\vdash AGp$（推广）。

可满足性

CTL 公式的可满足性是可判定的，其判定复杂性为 EXPTIME 完全。

§3.2.3 CTL 公式的不动点表示

给定一个 AP 上的 Kripke 结构 $M = \langle S, R, I, L\rangle$ 。给定的一个 CTL 公式 p 。 S 的子集 $\{s \in S \mid M, s \models p\}$ 记为 $[[p]]$ 。

定义函数 $ex : 2^S \rightarrow 2^S$ 如下：

$$ex(A) = \{s \in S \mid \exists s' \in S, (s, s') \in \Delta \wedge s' \in A\}$$

我们有以下等式：

$[[p]]$	$=$	$\{s \mid p \in L(s)\}$
$[[\neg p]]$	$=$	$S \setminus [[p]]$
$[[p \wedge q]]$	$=$	$[[p]] \cap [[q]]$
$[[EXp]]$	$=$	$ex([[p]])$
$[[EGp]]$	$=$	$[[p]] \cap ex([[EGp]])$
$[[E(pUq)]]$	$=$	$[[q]] \cup ([[p]] \cap ex([[E(pUq)]])$

由以上等式知 $[[EGp]]$ 和 $[[E(pUq)]]$ 分别是 $\tau_1(Z) = [[p]] \cap ex(Z)$ 和 $\tau_2(Z) = [[q]] \cup ([[p]] \cap ex(Z))$ 的不动点。 根据 CTL 语义知 τ_1 和 τ_2 有最小和最大不动点，且 $[[EGp]]$ 是 τ_1 的最大不动点， $[[E(pUq)]]$ 是 τ_2 的最小不动点，即

$$\begin{array}{lcl} [[EGp]] & = & \nu Z([[p]] \cap ex(Z)) \\ [[E(pUq)]] & = & \mu Z([[q]] \cup ([[p]] \cap ex(Z))) \end{array}$$

类似地，其它模态算子也可以用不动点刻画。为方便书写，我们直接将 p, q 看成 S 的子集，模态算子和逻辑联接符看成是 2^S 上的函数。

AFp	$=$	$\mu Z(p \vee AXZ)$
AGp	$=$	$\nu Z(p \wedge AXZ)$
EFp	$=$	$\mu Z(p \vee EXZ)$
EGp	$=$	$\nu Z(p \wedge EXZ)$
$A(pUq)$	$=$	$\mu Z(q \vee (p \wedge AXZ))$
$A(pRq)$	$=$	$\nu Z(q \wedge (p \vee AXZ))$
$E(pUq)$	$=$	$\mu Z(q \vee (p \wedge EXZ))$
$E(pRq)$	$=$	$\nu Z(q \wedge (p \vee EXZ))$

设 φ 为 CTL 公式， M 为 Kripke 结构。我们有

$$\begin{array}{lll} M, s \models \varphi & \text{当且仅当} & s \in [[\varphi]] \text{ 。} \\ M \models \varphi & \text{当且仅当} & I \subseteq [[\varphi]] \text{ 。} \end{array}$$

§3.2.4 CTL 公式在 Kripke 结构中的推理与交错 Büchi 自动机

由于 CTL 公式在状态上解释，给定 Kripke 结构 $M = \langle S, \Delta, I, L \rangle$ 和一个状态 s ，我们可以根据 s 或其后续状态的情况来判断 s 是否满足一个公式。

CTL 公式在 Kripke 结构中的推理

我们定义三种推理规则

$$\frac{\top}{A}, \quad \vee\frac{A_1 \quad \cdots \quad A_n}{A}, \quad \wedge\frac{A_1 \quad \cdots \quad A_n}{A}$$

分别表示不需要前提即可得出结论 A ，有 n 个前提中的一个前提即可得出结论 A ，有 n 个前提中的所有前提则可得出结论 A 。 我们考虑 CTL 公式的 NNF 范式。

$$\frac{\top}{s \vdash_M p} \quad p \in L(s) \qquad \frac{\top}{s \vdash_M \neg p} \quad p \in AP \setminus L(s)$$

$$\wedge\frac{s \vdash_M p \quad s \vdash_M q}{s \vdash_M p \wedge q} \qquad \vee\frac{s \vdash_M p \quad s \vdash_M q}{s \vdash_M p \vee q}$$

$$\wedge\frac{s_1 \vdash_M p \quad \cdots \quad s_n \vdash_M p}{s \vdash_M AXp} \quad (s_1, ..., s_n) = \{s' \mid (s, s') \in \Delta\}$$

$$\vee\frac{s_1 \vdash_M p \quad \cdots \quad s_n \vdash_M p}{s \vdash_M EXp} \quad (s_1, ..., s_n) = \{s' \mid (s, s') \in \Delta\}$$

$$\vee\frac{s \vdash_M q \quad s \vdash_M p \wedge AXA(pUq)}{s \vdash_M A(pUq)} \qquad \vee\frac{s \vdash_M q \quad s \vdash_M p \wedge EXE(pUq)}{s \vdash_M E(pUq)}$$

$$\wedge\frac{s \vdash_M q \quad s \vdash_M p \vee AXA(pRq)}{s \vdash_M A(pRq)} \qquad \wedge\frac{s \vdash_M q \quad s \vdash_M p \vee EXE(pRq)}{s \vdash_M E(pRq)}$$

对于一个向上的分枝, 如果该分枝终止于某一条规则, 推理成功当且仅当需要的条件是 $\top$; 如果该分枝不终止, 推理成功当且仅当该分枝中无限出现的公式中没有 EU 或 AU 类型的公式。

交错 Büchi 自动机

定义

$$\begin{array}{l} sf_0(A(bRc)) = \{b \vee AXA(bRc), AXA(bRc)\} \\ sf_0(A(bUc)) = \{b \wedge AXA(bUc), AXA(bUc)\} \\ sf_0(E(bRc)) = \{b \vee EXE(bRc), EXE(bRc)\} \\ sf_0(E(bUc)) = \{b \wedge EXE(bUc), EXE(bUc)\} \end{array}$$

定义 $sf(\varphi)$ 为公式的集合如下。

$a \in sf(\varphi)$ 当且仅当
a 在 φ 中出现或 a' 在 φ 中出现且 $a \in sf_0(a')$ 。

给定 Kripke 结构 $M = \langle S, \Delta, I, L\rangle$ 和公式 φ 。构造交错 Büchi 自动机 $\mathcal{B}(M, \varphi) = \langle \Sigma, S', \Delta', I', F\rangle$ 如下:

Σ	$=$	$\{a\}$
S'	$=$	$\{s \vdash_M \psi \mid s \in S, \psi \in sf(\varphi)\}$
$\Delta'(t,a)$	$=$	$\{\{t_1,...,t_n\}\}$ 对应 $\wedge$- 规则
$\Delta'(t,a)$	$=$	$\{\{t_1\},...,\{t_n\}\}$ 对应 $\vee$- 规则
$\Delta'(t,a)$	$=$	$\{\{\}\}$ 若 $t = \top$
$\Delta'(t,a)$	$=$	$\{\{t\}\}$ 若 t 是叶结点且 $t \neq \top$
I'	$=$	$\{s \vdash_M \varphi \mid s \in I\}$
F	$=$	$\{s \vdash_M E(pRq) \mid s \in S, E(pRq) \in sf(\varphi)\} \cup \{s \vdash_M A(pRq) \mid s \in S, A(pRq) \in sf(\varphi)\}$

则 $\mathcal{B}(M,\varphi)$ 可以看成是 $M \models \varphi$ 的证明结构。 $\mathcal{B}(M,\varphi)$ 为空当且仅当 $M \models \neg\varphi$ 。

对于任意给定 Kripke 结构 $\langle S, \Delta, I, L\rangle$ 和 NNF 范式的 CTL 公式 φ ，我们能够构造一个相应的交错 Büchi 自动机 $\langle \Sigma, S', \Delta', I', F\rangle$ 且 $|S'| \leq |S| \cdot |sf(\varphi)|$ 。

§3.2.5 计算树逻辑 CTL*

由于 CTL 中描述分枝情况和描述状态的前后关系的算子成对出现，在一定程度上限制了 CTL 的表达能力。我们可以将其拆开使用。这个逻辑记作 CTL* 。给定一个原子命题集合 AP 。CTL* 公式的集合由以下语法给出。

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid X\,\phi \mid F\,\phi \mid G\,\phi \mid \phi\,U\,\phi \mid \phi\,R\,\phi \mid A\phi \mid E\phi$$

其中 p 为 AP 中任意命题。我们可以将 CTL* 中的公式分为状态公式和路径公式。

若 $p \in AP$,	则 p 是状态公式
若 φ 和 ψ 是状态公式,	则 $\neg\varphi, \varphi \vee \psi, \varphi \wedge \psi$ 是状态公式
若 φ 是路径公式,	则 $E\varphi$ 和 $A\varphi$ 是状态公式
若 φ 是状态公式,	则 φ 是路径公式
若 φ 和 ψ 是路径公式,	则 $\neg\varphi, \varphi \vee \psi, \varphi \wedge \psi, X\varphi, F\varphi, G\varphi, \varphi U\psi, \varphi R\psi$ 是路径公式

CTL* 公式在一个树状结构或 Kripke 结构上解释。设 $M = \langle S, R, I, L\rangle$ 是原子命题集 AP 上的 Kripke 结构。 CTL* 公式的语义如下：

$M,s \models p$	若 $p \in AP$ 且 $p \in L(s)$
$M,s \models \neg\varphi$	若 $M,s \not\models \varphi$
$M,s \models \varphi \vee \psi$	若 $M,s \models \varphi$ 或 $M,s \models \psi$
$M,s \models \varphi \wedge \psi$	若 $M,s \models \varphi$ 且 $M,s \models \psi$
$M,s \models A\varphi$	若对于所有 M 中以 s 为起点的路径 π ： $M,\pi \models \varphi$
$M,s \models E\varphi$	若存在 M 中以 s 为起点的路径 π ： $M,\pi \models \varphi$
$M,\pi \models p$	若 $p \in AP$ 且 $M,\pi_0 \models p$
$M,\pi \models A\varphi$	若 $M,\pi_0 \models A\varphi$
$M,\pi \models E\varphi$	若 $M,\pi_0 \models E\varphi$
$M,\pi \models \neg\varphi$	若 $M,\pi \not\models \varphi$
$M,\pi \models \varphi \vee \psi$	若 $M,\pi \models \varphi$ 或 $M,\pi \models \psi$
$M,\pi \models \varphi \wedge \psi$	若 $M,\pi \models \varphi$ 且 $M,\pi \models \psi$
$M,\pi \models X\varphi$	若 $M,\pi^1 \models \varphi$
$M,\pi \models G\psi$	若 $\forall i \geq 0, M,\pi^i \models \psi$
$M,\pi \models F\varphi$	若 $\exists i \geq 0, M,\pi^i \models \varphi$
$M,\pi \models \varphi U \psi$	若 $\exists i \geq 0, M,\pi^i \models \psi$ 且 $\forall 0 \leq j < i, M,\pi^j \models \varphi$
$M,\pi \models \varphi R \psi$	若 $\forall i \geq k$, 若 $\forall 0 \leq j < i, M,\pi^j \not\models \varphi$ 则 $M,\pi^i \models \psi$

设 φ 为 CTL* 状态公式， M 为 Kripke 结构。

$$M \models \varphi$$

当且仅当

$$\text{对所有 } s \in I \text{ ， } M,s \models \varphi \text{ 。}$$

PLTL 和 CTL 是 CTL* 的子集。设 PLTL 、 CTL 和 CTL* 都在 Kripke 结构上解释。一个 PLTL 公式 φ 等价于一个 CTL* 公式 $A\varphi$ 。这样 PLTL 公式就可以看成是状态公式。 在这样的解释下，我们可以比较 PLTL 、 CTL 和 CTL* 的状态公式。

PLTL 和 CTL 互不录属且 CTL* 大于 PLTL 和 CTL 的并集。

CTL 公式 $AGEFp$ 不能用 PLTL 表示。
PLTL 公式 $F(p \wedge Xp)$ 不能用 CTL 表示。
CTL* 公式 $E(GFp)$ 不能用 CTL 表示，不能用 PLTL 表示。

§3.2.6 μ- 演算

由关于 CTL 的讨论我们知道 CTL 公式可以用模态算子 AX,EX, 命题变量和不动点表示。在这种表示中，命题变量可以出现在 AX, EX, μ, ν 之后。将这个约束去掉，我们可以得到一个表达能力更强的逻辑。称为 μ- 演算。 给定一个原子命题集合 AP ，一个变量集合 V 。一种简单的 μ- 演算的公式的集合可由以下语法给出。

$$\phi ::= p \mid X \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle . \rangle \phi \mid [.]\phi \mid \mu X.\phi \mid \nu X.\phi$$

其中 p 为 AP 中任意命题， X 为变量， $\mu X.\phi$ 和 $\nu X.\phi$ 的 ϕ 中不受囿的 X 必须在偶数个 $\neg$ 符号的作用范围之下。

带有动作描述的 μ- 演算

对前述简单 μ- 演算做扩充，增加动作的描述，我们可以得到一种能够描述动作的时序关系的时序逻辑。 给定一个原子命题集合 AP ，一个变量集合 V ，和一个动作集合 A 。 μ- 演算公式的集合由以下语法给出。

$$\phi ::= p \mid X \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \langle a\rangle\phi \mid [a]\phi \mid \mu X.\phi \mid \nu X.\phi$$

其中 p 为 AP 中任意命题， X 为变量， a 为动作，并且 $\mu X.\phi$ 和 $\nu X.\phi$ 的 ϕ 中不受囿的 X 必须在偶数个 $\neg$ 符号的作用范围之下。

μ- 演算公式在 LTS 上解释。设 $M = \langle \Sigma, S, \Delta, I, L\rangle$ 为原子命题集 AP 上的 LTS ， $e : V \to 2^S$ 为变量到 S 子集的赋值。 μ- 演算公式的语义如下：

$$\begin{array}{lcl}
[[p]]e & = & \{s \mid p \in L(s)\} \\
[[X]]e & = & e(X) \\
[[\neg\phi]]e & = & S \setminus [[\phi]]e \\
[[\phi_1 \wedge \phi_2]]e & = & [[\phi_1]]e \cap [[\phi_2]]e \\
[[\phi_1 \vee \phi_2]]e & = & [[\phi_1]]e \cup [[\phi_2]]e \\
[[\langle a\rangle\phi]]e & = & \{s \mid \exists s'.s \xrightarrow{a} s' \wedge s' \in [[\phi]]e\} \\
[[[a]\phi]]e & = & \{s \mid \forall s'.s \xrightarrow{a} s' \Rightarrow s' \in [[\phi]]e\} \\
[[\mu X.\phi]]e & = & \cap\{S' \subseteq S \mid [[\phi]]e[X/S'] \subseteq S'\} \\
[[\nu X.\phi]]e & = & \cup\{S' \subseteq S \mid S' \subseteq [[\phi]]e[X/S']\}
\end{array}$$

所有变量都是受囿变量的公式称为闭公式。闭公式的语义不受 e 的影响。设 M 为 LTS ， φ 为闭公式。 φ 在 M 中的语义可写为 $[[\varphi]]$ 。

$$M \models \varphi$$

当且仅当

$$I \subseteq [[\varphi]] \text{ 。}$$

模态算子的对偶关系与 NNF 范式

逻辑连接符 $\neg$ 只出现在命题前面的公式称为 NNF 范式。每个 μ- 演算公式等价于一个 μ- 演算的 NNF 范式。一个 μ- 演算的 NNF 范式可应用以下对偶关系构造。

$$\begin{array}{lcl}
[a]\phi & \equiv & \neg\langle a\rangle\neg\phi \\
\mu X.\phi & \equiv & \neg\nu X.\neg\phi[X/\neg X]
\end{array}$$

μ 和 ν 的递推关系

关于 μ 和 ν ，我们有以下等式。

$$\begin{array}{lcl}
\mu X.\phi & \equiv & \phi_X^{\mu X.\phi} \\
\nu X.\phi & \equiv & \phi_X^{\nu X.\phi}
\end{array}$$

μ- 演算公式在 Kripke 结构中的推理

与 CTL 公式类似，给定一个 Kripke 结构 M 和一个状态 s ，我们可以根据 s 或其后续状态的情况来判断 s 是否满足一个（ NNF 范式）的 μ- 演算公式。

$$\frac{\top}{s \vdash_M p} \quad p \in L(s) \qquad \frac{\top}{s \vdash_M \neg p} \quad p \in AP \setminus L(s)$$

$$\wedge\frac{s \vdash_M p \quad s \vdash_M q}{s \vdash_M p \wedge q} \qquad \vee\frac{s \vdash_M p \quad s \vdash_M q}{s \vdash_M p \vee q}$$

$$\vee\frac{s_1 \vdash_M p \quad \cdots \quad s_n \vdash_M p}{s \vdash_M \langle a \rangle p} \quad \{s_1, ..., s_n\} = \{s' \mid (s, a, s') \in \Delta\}$$

$$\wedge\frac{s_1 \vdash_M p \quad \cdots \quad s_n \vdash_M p}{s \vdash_M [a] p} \quad \{s_1, ..., s_n\} = \{s' \mid (s, a, s') \in \Delta\}$$

$$\frac{s \vdash_M \varphi_X^{\mu X.\varphi}}{s \vdash_M \mu X.\varphi} \qquad \frac{s \vdash_M \varphi_X^{\nu X.\varphi}}{s \vdash_M \nu X.\varphi}$$

对于一个向上的分枝, 如果该分枝终止于某一条规则, 推理成功当且仅当需要的条件是 $\top$；如果该分枝不终止，推理成功当且仅当在该分枝中无限出现的公式中每个 $\mu X.\varphi$ 类型的公式都是该分枝中某个 $\nu X.\psi$ 类型公式的子公式。

CTL 公式到 μ- 演算公式的转换

由于 CTL 公式不牵涉到动作的描述，可以假定 CTL 所描述系统中，只用同一个动作，记作 a 。这样，可以用以下规则将 CTL 公式转换到 μ- 演算公式。

$$\begin{aligned}
T(p) &= p \\
T(\neg\varphi) &= \neg T(\varphi) \\
T(\varphi \wedge \psi) &= T(\varphi) \wedge T(\psi) \\
T(EX\varphi) &= \langle a \rangle T(\varphi) \\
T(E(\varphi U \psi)) &= \mu Y.(T(\psi) \vee (T(\varphi) \wedge \langle a \rangle Y)) \\
T(EG\varphi) &= \nu Y.(T(\varphi) \wedge \langle a \rangle Y)
\end{aligned}$$

§3.3 交错时序逻辑 (ATL)

交错时序逻辑是一种能够描述动作主体合作完成任务的逻辑.给定一个原子命题集合 AP 。一个动作主体的集合 Σ 。 ATL 公式的集合由以下语法给出。

$$\phi ::= \top \mid \bot \mid p \mid \neg\phi \mid \phi \vee \phi \mid \langle\langle A \rangle\rangle O\phi \mid \langle\langle A \rangle\rangle \Box\phi \mid \langle\langle A \rangle\rangle \phi U \phi$$

其中 p 为 AP 中任意命题、 $A \subseteq \Sigma$ 为 Σ 的子集。

ATL 公式在 ATS 上解释。设 $M = \langle \Sigma, S, \Delta, I, L \rangle$ 为 AP 上的 ATS 。给定 $a \in \Sigma$ 。一个 a 的策略 f_a 是 $S^+ \to 2^S$ 的一个函数，满足对所有 $\lambda \in S^*$ 和所有 $s \in S$ ，

$$f(\lambda \cdot s) \in \Delta(s, a)$$

给定 $A \subseteq \Sigma$ ， $F_A = \{f_a \mid a \in A\}$ ，在状态 s 应用策略 F_A 的结果 $out(s, F_A)$ 满足条件：

$[s_i]_{i \geq 0} \in out(s, F_A)$ 当前仅当 $s = s_0$ 且对所有 i, $s_{i+1} \in \cap_{a \in A} f_a(s_0 s_1 \cdots s_i)$ 。

ATL 公式的语义如下：

$M, s \models p$	若 $p = \top$ ，或 $p \in AP$ 且 $p \in L(s)$
$M, s \models \neg\varphi$	若 $M, s \not\models \varphi$
$M, s \models \varphi \vee \psi$	若 $M, s \models \varphi$ 或 $M, s \models \psi$
$M, s \models \langle\langle A\rangle\rangle O\varphi$	若存在 F_A 且对所有 $\pi \in out(s, F_A)$ ， $M, \pi_1 \models \varphi$
$M, s \models \langle\langle A\rangle\rangle \Box\varphi$	若存在 F_A 且对所有 $\pi \in out(s, F_A)$ ，对所有 $i \geq 0$ ， $M, \pi_i \models \varphi$
$M, s \models \langle\langle A\rangle\rangle \varphi U \psi$	若存在 F_A 且对所有 $\pi \in out(s, F_A)$ ， 存在 $i \geq 0$ ， $\pi_i \models \psi$ 且对所有 $0 \leq j < i$ ， $M, \pi_j \models \varphi$

除了以上基本算子外，还可以定义其它算子，如：

$\Diamond\varphi$	$=$	$\top U \varphi$
$[[A]]O\varphi$	$=$	$\neg\langle\langle A\rangle\rangle O\neg\varphi$
$[[A]]\Box\varphi$	$=$	$\neg\langle\langle A\rangle\rangle \Diamond\neg\varphi$

§3.4 说明

本章主要介绍程序逻辑。线性时序逻辑部分可参考 [Peled01] 。一阶线性时序逻辑部分可参考 [Peled01, MP83] 。线性 μ- 演算部分可参考 [Kaivola95, BEM96, DHL06] 。分枝时序逻辑部分可参考 [CGP99, Peled01] 。模态 μ- 演算部分可参考 [CGP99, BC96] 。交错时序逻辑部分部分可参考 [AHK97] 。

参考文献

[AHK97] Rajeev Alur, Thomas A. Henzinger and Orna Kupferman. Alternating-Time Temporal Logic. COMPOS 1997: 23-60.

[BC96] Girish Bhat and Rance Cleaveland. Efficent Local Model-Checking for Fragments of the Modal mu-Calculus. TACAS 1996: 107-126.

[BEM96] Julian Bradfield, Javier Esparza and Angelika Mader. An Effective Tableau System for the Linear Time μ-Calculus. ICALP'96. LNCS 1099:98-109.

[CGP99] E. Clark, O. Grumberg and D. Peled. Model Checking. MIT press, 1999.

[DHL06] Christian Dax, Martin Hofmann and Martin Lange. A Proof System for the Linear Time μ-Calculus . Lecture Notes in Computer Science 4337 (FSTTCS 2006):273-284.

[Kaivola95] Roope Kaivola. Axiomatising Linear Time μ-Calculus. CONCUR'95. LNCS 962:423-437.

[MP83] Zohar Manna and Amir Pnueli. How to cool a temporal proof system for your pet language. The 10th ACM Symposium on Principles of Programming Languages 141-154. 1983.

[Peled01] Doron A. Peled. Software Reliability Methods. Springer-Verlag. 2001.

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，我们可以应用推理方法。本章先介绍以一类时序逻辑公式为程序性质描述语言的推理方法，再介绍以断言为程序性质描述语言的、针对特定类型模型的推理方法。

§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描述语言的推理方法。 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 $\{p\}t\{q\}$ 在解释 I 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 $\models_I\{\varphi\}S\{\psi\}$ 当且仅当对所有 $t\in S$, $\models_I\{\varphi\}t\{\psi\}$ 成立。 在 I 给定的情况下, $\models_I\{\varphi\}S\{\psi\}$ 也记作 $\varphi\to[S]\psi$ 。我们有：

$$\frac{\varphi\to[S_0]\psi \qquad \varphi\to[S_1]\psi}{\varphi\to[S_0\cup S_1]\psi}$$

§4.1.1 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 $E(t)=p$ 和 $E(T)=\bigvee_{t\in T}E(t)$ 。若 $E(T)$ 成立，则 T 中有可执行的迁移。

- 迁移公理：

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

- Θ 规则：

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

- O 规则：

$$\frac{\varphi \Rightarrow (\psi \vee E(T)) \quad \varphi \to [T]\psi}{\varphi \Rightarrow O\psi}$$

- 推断规则：

$$\frac{\varphi' \Rightarrow \varphi \quad \varphi \to [T]\psi \quad \psi \Rightarrow \psi'}{\varphi' \to [T]\psi'}$$

§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：

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

- R 规则：

$$\frac{\zeta \Rightarrow \varphi' \quad (\varphi' \wedge \neg\psi) \to [T]\varphi' \quad \varphi' \Rightarrow \varphi}{\zeta \Rightarrow \psi R \varphi}$$

- $\Box$ 规则：

$$\frac{\zeta \Rightarrow \varphi' \quad \varphi' \to [T]\varphi' \quad \varphi' \Rightarrow \varphi}{\zeta \Rightarrow \Box\varphi}$$

- 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

$$\frac{\varphi \Rightarrow (\psi \vee \zeta) \quad \zeta \Rightarrow (\zeta_0 \wedge w_x^e \wedge (\psi \vee E(T))) \quad (\zeta \wedge e = v) \to [T](\psi \vee (\zeta \wedge e \sqsubset v))}{\varphi \Rightarrow \zeta_0 U \psi}$$

- $\Diamond$ 规则：

$$\frac{\varphi \Rightarrow (\psi \vee \zeta) \quad \zeta \Rightarrow (w_x^e \wedge (\psi \vee E(T))) \quad (\zeta \wedge e = v) \to [T](\psi \vee (\zeta \wedge e \sqsubset v))}{\varphi \Rightarrow \Diamond\psi}$$

§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 $t : p \to (x_1, ..., x_k) = (e_1, ..., e_k)$ 。定义 $\bar{x}(t) = (x_1, ..., x_k, x_{k+1}, ..., x_n)$ ， $\bar{e}(t) = (e_1, ..., e_k, x_{k+1}, ..., x_n)$ 。我们有以下推理规则。

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

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) 。

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

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 $\models_I \Phi \to \varphi$ 。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 $\models_{\mathcal{I}}\{p\}\rho\{q\}$ 。 在 I 给定的情况下， $\models_I\{p\}\rho\{q\}$ 也记作 $p\to[\rho]q$ 。

§4.2.1 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：
$$\forall v_1',...,v_n'.(\rho\to\psi^{v_1',...,v_n'}_{v_1,...,v_n})\to[\rho]\psi$$
- Θ 规则：
$$\frac{\Theta\Rightarrow\varphi}{\varphi}$$
- O 规则：
$$\frac{\varphi\Rightarrow(\psi\vee\exists v_1'...v_n'.\rho)\quad \varphi\to[\rho]\psi}{\varphi\Rightarrow O\psi}$$
- 推断规则：
$$\frac{\varphi'\Rightarrow\varphi\quad \varphi\to[\rho]\psi\quad \psi\Rightarrow\psi'}{\varphi'\to[\rho]\psi'}$$

§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：

$$\frac{\varphi \wedge \rho \to \psi^{v_1',\ldots,v_n'}_{v_1,\ldots,v_n}}{\varphi \to [\rho]\psi}$$

- R 规则：

$$\frac{\begin{array}{l}\zeta \Rightarrow \varphi' \\ (\varphi' \wedge \neg\psi) \to [\rho]\varphi' \\ \varphi' \Rightarrow \varphi\end{array}}{\zeta \Rightarrow \psi R \varphi}$$

- $\Box$ 规则：

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

- 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

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee \zeta) \\ \zeta \Rightarrow (\zeta_0 \wedge w_x^e \wedge (\psi \vee \exists v_1' \ldots v_n'.\rho)) \\ (\zeta \wedge e = v) \to [\rho](\psi \vee (\zeta \wedge e \sqsubset v))\end{array}}{\varphi \Rightarrow \zeta_0 U \psi}$$

- $\Diamond$ 规则：

$$\frac{\begin{array}{l}\varphi \Rightarrow (\psi \vee \zeta) \\ \zeta \Rightarrow (w_x^e \wedge (\psi \vee \exists v_1' \ldots v_n'.\rho)) \\ (\zeta \wedge e = v) \to [\rho](\psi \vee (\zeta \wedge e \sqsubset v))\end{array}}{\varphi \Rightarrow \Diamond\psi}$$

§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 $T \in \mathcal{L}^{(B,V)}_{\hookrightarrow}$ 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.

中国科学院研究生院课程讲稿

§5 模型检测方法

要证明一个程序或程序模型是否具备某些性质，我们可以应用模型检测方法。模型检测主要有三类方法：一类基于状态的分析，一类基于路径的分析，另外针对程序模型状态数量大这个瓶颈问题，基于时序逻辑限界语义的模型检测方法是可能缓解这个问题的一种图径。本章介绍这三类模型检测方法。

§5.1 基于状态分析的模型检测

基于状态的分析的出发点是计算模型的哪些状态满足哪些性质。基于状态分析的模型检测适用于 CTL 性质。一个系统的正确性依赖于系统可达的状态的性质。每个 CTL 公式都是状态公式。给定一个模型，我们可以判断其中的任何状态是否满足这个公式。

§5.1.1 状态标号算法

由于 $\{\neg, \vee\}$ 构成命题逻辑联结词的完全集，我们只考虑这些命题逻辑联结词。由于 $\{EX, EG, EU\}$ 构成 CTL 模态算子的完全集，我们只考虑这些模态算子。

给定 AP 上的 Kripke 结构 $\langle S, R, I, L\rangle$ 和一个 CTL 公式 φ，该标号算法的主要思想是逐步扩充 L 使得 $L(s)$ 包含每个在 s 满足的 φ 的子公式。

假设每个状态是否满足 φ 的子公式已经计算过。那么为计算每个状态是否满足 φ 我们有以下情况。

（ 1 ）若 $\varphi = p$ 且 p 是命题，则对每个 s， p 是否在 $L(s)$ 已经有系统给定，无需计算。

（ 2 ）若 $\varphi = \neg\varphi_0$，则对每个 s，若 $\varphi_0 \notin L(s)$，则将 φ 加入 $L(s)$。

（ 3 ）若 $\varphi = \varphi_0 \vee \varphi_1$，则对每个 s，若 $\varphi_0 \in L(s)$ 或 $\varphi_1 \in L(s)$，则将 φ_0 加入 $L(s)$。

（ 4 ）若 $\varphi = EX\varphi_0$，则对每个 s，若存在 t 使得 $R(s,t)$ 且 $\varphi_0 \in L(t)$，则将 φ 加入 $L(s)$。

（ 5 ）若 $\varphi = E(\varphi_0 U \varphi_1)$，则

（ 5a ）对每个 s，若 $\varphi_1 \in L(s)$，则将 φ 加入 $L(s)$。

（ 5b ）对每个 s，若存在 t 使得 $R(s,t)$ 且 $\varphi \in L(t)$，且 $\varphi_0 \in L(s)$，则将 φ 加入 $L(s)$。

（ 5c ）重复 (5b) 直至 L 不起变化。

（ 6 ）若 $\varphi = EG\varphi_0$，则

（ 6a ）计算 $S' = \{s \mid \varphi_0 \in L(s)\}$。

（ 6b ）计算 $\langle S, R\rangle$ 的导出子图 $\langle S', R'\rangle$ 的非平凡强连通分图的顶点集合 S''，并对每个 $s \in S''$，将 φ 加入 $L(s)$。

（ 6c ）对每个 $s \in S'$，若存在 t 使得 $R(s,t)$ 且 $\varphi \in L(t)$，则将 φ 加入 $L(s)$。

（ 6d ）重复 (6c) 直至 L 不起变化。

然后我们可以直接根据已经计算出的 L 求得系统的初始状态是否满足 φ，即

$$M, s \models \varphi \quad \text{当且仅当} \quad \varphi \in L(s)。$$
$$M \models \varphi \quad \text{当且仅当} \quad \varphi \in \bigcap_{s\in I} L(s)。$$

§5.1.2 符号模型检测原理

符号模型检测是建立在对状态集合的操作上。主要是计算一个公式在哪些状态上成立。基于这个思路，一个公式可以看作是满足这个公式的系统状态的集合。一个系统满足某个状态公式，就是说系统的初始状态包含于这个公式所代表的状态集合。符号模型检测的基本思想是用 CTL 公式的不动点刻画以及用命题逻辑公式或相关结构表示状态集合并进行相应的运算。

符号模型

记 $\Phi(V)$ 为自由命题变量集合是 V 的子集的命题逻辑公式的集合。若 V 为变量集合，则 V' 为变量集合，定义如下： $v' \in V'$ 当且仅当 $v \in V$ 。

Definition 5.1 给定一个命题集合 AP 。一个 AP 上的符号模型是一个四元组

$$\langle V, R, \Theta, N \rangle$$

其中 V 为命题变量集合， R 为 $V \cup V'$ 上的公式， Θ 为 V 上的公式， $N : AP \to \Phi(V)$ 为 AP 到 V 上公式集合 $\Phi(V)$ 的映射。

系统状态由变量 $x_1, ..., x_n$ 决定。一个系统状态用 n 元组 $(a_1, ..., a_n)$ 表示。 $(a_1, ..., a_n) \models q$ 当且仅当 $q_{x_1,...,x_n}^{a_1,...,a_n} = 1$ 。

系统状态的集合为 $\{0,1\}^n$ 。迁移关系由 $R(x_1, ..., x_n, x'_1, ..., x'_n)$ 决定。系统的初始状态集合由 $\Theta(x_1, ..., x_n)$ 决定。设 $AP = \{p_1, ..., p_k\}$ 。满足 p_i 的状态由 $N(p_i)$ 决定。

路径

我们用 $s \to s'$ 表示存在从 s 到 s' 的迁移，即 $(s, s') \models R$ 。符号模型 $\langle V, R, \Theta, N \rangle$ 上的一条路径是状态集 $\{0,1\}^n$ 上的一个无穷序列

$$s_0 s_1 s_2 ...$$

其中对任意 $i \geq 0$, $s_i \to s_{i+1}$ 。

运行

符号模型 $\langle V, R, \Theta, N \rangle$ 上的一次运行是该符号模型上的第一个状态满足 Θ 的一条路径。

符号模型与 Kripke 结构

给定符号模型 (V, R, Θ, N) 。该模型对应于 $AP = \{p_1, ..., p_k\}$ 上的 Kripke 结构 $M = \langle S, \Delta, I, L \rangle$ 其中 S, Δ, I, L 定义如下：

$$\begin{aligned}
&S = \{(a_1, ..., a_n) \mid a_i \in \{0,1\}\} \\
&\Delta = \{((a_1, ..., a_n), (a'_1, ..., a'_n)) \mid (a_1, ..., a_n, a'_1, ..., a'_n) \models R(V, V')\} \\
&I = \{(a_1, ..., a_n)) \mid (a_1, ..., a_n) \models \Theta\} \\
&L((a_1, ..., a_n)) = \{p_i \in AP \mid (a_1, ..., a_n) \models N(p_i)\}
\end{aligned}$$

Kripke 结构与符号模型

给定 Kripke 结构 $M = \langle S, \Delta, I, L \rangle$ 。我们可以将 S 编号。若 S 有 $\leq 2^n$ 个元素，则可以用 n 个命题 $x_1, ..., x_n$ 来表示。 S 中的第 i 个元素记为 $s(i-1)$ 。

用 $(a_1, ..., a_n)$ 表示 $x_1, ..., x_n$ 取值的 n 元组。则该 n 元组表示 $s(a_1 \cdot 2^n + \cdots + a_{n-1} \cdot 2 + a_n)$ 。

一个 $x_1, ..., x_n$ 上的公式表示一个集合，即满足该公式的 $x_1, ..., x_n$ 取值的 n 元组的集合。

状态的转化关系的集合一样可以用公式表示。我们需引进 n 个新变量，记作 $x'_1, ..., x'_n$ 。用 $(a_1, ..., a_n, a'_1, ..., a'_n)$ 表示 $x_1, ..., x_n, x'_1, ..., x'_n$ 取值的 $2n$ 元组。则该 $2n$ 元组表示 $(s(a_1 \cdot 2^n + \cdots + a_{n-1} \cdot 2 + a_n), s(a'_1 \cdot 2^n + \cdots + a'_{n-1} \cdot 2 + a'_n)) \in \Delta$ 。

一个 $x_1, ..., x_n, x'_1, ..., x'_n$ 上的公式表示一个迁移集合，即满足该公式的 $a_1, ..., a_n, a'_1, ..., a'_n$ 取值的 $2n$ 元组的集合。

对于标号函数 $L : S \to 2^{AP}$，我们定义 $N : AP \to 2^S$ 如下：$s \in N(p)$ 当且仅当 $p \in L(s)$。则 L 和 N 作为系统模型中描述状态满足哪些基本命题的作用是等价的。由于 $N(p)$ 是一个状态集合，$N(p)$ 可以用一个公式表示。

给定 $AP = \{p_1, ..., p_k\}$ 上的 Kripke 结构 $M = \langle S, \Delta, I, L\rangle$。该模型对应于符号模型 (V, R, Θ, N)，其中 V, R, Θ, N 定义如下：

$$
\begin{aligned}
&V = \{a_1, ..., a_n\}\\
&(a_1, ..., a_n, a'_1, ..., a'_n) \models R(V, V') \text{ 当且仅当 } (s(a_1, ..., a_n), s(a'_1, ..., a'_n)) \in \Delta\\
&(a_1, ..., a_n) \models \Theta \text{ 当且仅当 } s(a_1, ..., a_n) \in I\\
&N(p_i) = \Theta_i \text{ 且 } \Theta_i \text{ 定义如下：} (a_1, ..., a_n) \models \Theta_i \text{ 当且仅当 } p_i \in L(s(a_1, ..., a_n))
\end{aligned}
$$

符号模型检测

一个 (V, R, Θ, N) 表示的系统是否满足一个时序逻辑公式，即系统所对应的 Kripke 结构 M 是否满足一个时序逻辑公式。

定义 $\exists x.\varphi = (\varphi^0_x \vee \varphi^1_x)$。定义函数 $ex : \Phi(V) \to \Phi(V)$ 如下：

$$ex(\varphi) = \exists x'_1 ... x'_n.(R(x_1, ..., x_n, x'_1, ..., x'_n) \wedge \varphi^{x'_1,...,x'_n}_{x_1,...,x_n})$$

用 $[[q]]$ 表示满足 q 的状态的集合。我们分 6 种情况计算一个公式 q 在 M 中的哪些状态上成立。

(1) $q = p_i$ 是原子命题，	则 $[[q]] = N(p_i)$
(2) q 是 $\neg q_0$，	则 $[[q]] = \neg[[q_0]]$
(3) q 是 $q_0 \vee q_1$，	则 $[[q]] = [[q_0]] \vee [[q_1]]$
(4) q 是 EXq_0，	则 $[[q]] = ex([[q_0]])$
(5) q 是 EGq_0，	则 $[[q]] = \nu Z.([[q_0]] \wedge ex(Z))$
(6) q 是 $E(q_0 U q_1)$，	则 $[[q]] = \mu Z.([[q_1]] \vee ([[q_0]] \wedge ex(Z)))$

我们有

$$M \models \varphi \quad \text{当且仅当} \quad \Theta \to [[\varphi]]$$

§5.1.3 二元决策图

Definition 5.2 给定一个变量集合 V。一个 V 上的二元决策图 *(BDD)* 是一个四元组

$$\langle N, n_0, E, L\rangle$$

其中 N 为节点的集合；$E : N \to N \times N$ 为定义节点出边的偏函数，若 $E(n) = \langle n', n''\rangle$，则称 $\langle n, n'\rangle$ 为 n 的 *0* 边，$\langle n, n''\rangle$ 为 n 的 *1* 边；n_0 为图的初始点；$L : N \to V \cup \{0, 1\}$ 为点的标号函数，满足 $L(n) \in \{0, 1\}$ 当且仅当 $E(N)$ 没有定义。

有序二元决策图 (OBDD)

给定一个 V 上变量的排序 $x_1, ..., x_m$，即 $x_1 < x_2 < \cdots < x_m$。约定 $x_i < 0$ 且 $x_i < 1$。一个二元决策图 $\langle N, n_0, E, L\rangle$ 是有序的，当且仅当 $E(a) = \langle b, c\rangle$ 则 $L(a) < L(b)$ 且 $L(a) < L(c)$。

给定 V 上的一个公式 φ。该公式可以用一个有序二元决策图 $\langle N, n_0, E, L\rangle$ 表示，其构造如下。

1. $N=\{n_0\}, L(n_0)=x_1, L'(n_0)=\varphi$.
2. 对任意 $n\in N$ 且 $E(n)$ 尚无定义，
若 $L(n)=x_i$ ，则在 N 中添加两个新的点 n', n'' 且令 $E(n)=\langle n', n''\rangle$ 。
若 $i<m$ ， 则令 $L(n')=L(n'')=x_{i+1}, L'(n')=(L'(n))^0_{x_i}, L'(n'')=(L'(n))^1_{x_i}$ 。
若 $i=m$ ， 则令 $L(n')=L'(n')=(L'(n))^0_{x_m}, L(n'')=L'(n'')=(L'(n))^1_{x_m}$ 。

若两个公式是等价的，则它们的这样构造的有序二元决策图是同构的。

最简有序二元决策图 (ROBDD)

以上构造的 OBDD 的结构和大小相当于一个真值表，有很多冗余部分。二元决策图化简的规则如下：

u 和 v 是叶节点， $L(u)=L(v)$ ，则合并这两个节点。
u 和 v 不是叶节点， $E(u)=E(v)$ 且 $L(u)=L(v)$ ，则合并这两个节点。
u 和 v 不是叶节点， $E(v)=\langle a, a\rangle$ ，则删除 v 点，将指向 v 的边 (或 n_0) 指到 a 。

将冗余部分用以上规则化简直到不能再简化时得到的决策图称为最简有序二元决策图。若两个公式是等价的，则它们的最简有序二元决策图是同构的。用最简有序二元决策图表示的公式的等价性容易判定。若一个公式为永真式，则该公式的 ROBDD 为只有一个叶节点的图且该叶节点的值为 1 。若一个公式为永假式，则该公式的 ROBDD 为只有一个叶节点的图且该叶节点的值为 0 。

变量排序

变量排序对ROBDD 的大小有很大的影响,对于一些问题,不同的变量排序所产生的ROBDD 的大小的区别是指数的。

设 $\varphi=\bigvee_{i=1}^{n}(x_{2i-1}\wedge x_{2i})$ 。给定变量的排序 $x_1, x_3, ..., x_{2n-1}, x_2, x_4, ..., x_{2n}$ 。则 φ 的 ROBDD 节点数为 2^{n+1} 。给定变量的排序 $x_1, x_2, ..., x_{2n}$ 。则 φ 的 ROBDD 节点数为 $2(n+1)$ 。 由于寻找最佳排序是 NP 难的问题。可用启发式的算法计算一个较好的变量排序。

有序二元决策图和布尔函数

一个公式可以看成是一个布尔函数，因此一个有序二元决策图对应于一个布尔函数。

用 $F|_N$ 表示 F 将定义域限制在 N 的子函数。定义 $\pi_i(\langle a_0, a_1\rangle)=a_i$ 。用 E^i_x 表示满足以下性质的通过修改 E 得到的函数 E' ：

若 $L(\pi_0(E(n)))\neq x$ 且 $L(\pi_1(E(n)))\neq x$	则 $E'(n)=E(n)$
若 $L(\pi_0(E(n)))=x$	则 $\pi_0(E'(n))=\pi_i(E(\pi_0(E(n))))$
若 $L(\pi_1(E(n)))=x$	则 $\pi_1(E'(n))=\pi_i(E(\pi_1(E(n))))$

用 $f_{OBDD}(x_1, ..., x_n)$ 表示与布尔函数 $f(x_1, ..., x_n)$ 对应的有序二元决策图。设 $f_{OBDD}(x_1, ..., x_n)=\langle N, n_0, E, L\rangle$ 且 $x_1<\cdots<x_n$ 。 则

$f_{OBDD}(0, x_2, ..., x_n)=\langle N', n_0', E', L'\rangle$ 其中
N' 是由 $\pi_0(E(n_0))$ 可达的所有节点的集合； $n_0'=\pi_0(E(n_0))$ ； $E'=E|_{N'}$ ； $L'=L|_{N'}$ 。
$f_{OBDD}(x_1, ..., x_{i-1}, 0, x_{i+1}, x_n)=\langle N', n_0, E', L'\rangle$ 其中
N' 是由 n_0 通过 $E^0_{x_i}$ 可达的所有节点的集合； $E'=E^0_{x_i}|_{N'}$ ； $L'=L|_{N'}$ 。

$f_{OBDD}(1, x_2, ..., x_n)=\langle N', n_0', E', L'\rangle$ 其中
N' 是由 $\pi_1(E(n_0))$ 可达的所有节点的集合； $n_0'=\pi_1(E(n_0))$ ； $E'=E|_{N'}$ ； $L'=L|_{N'}$ 。
$f_{OBDD}(x_1, ..., x_{i-1}, 1, x_{i+1}, x_n)=\langle N', n_0, E', L'\rangle$ 其中
N' 是由 n_0 通过 $E^1_{x_i}$ 可达的所有节点的集合； $E'=E^1_{x_i}|_{N'}$ ； $L'=L|_{N'}$ 。

量词消去

设 f,b,c 是 $OBDD$ 。设 n 是 $OBDD$ 的节点。用 $top(f)$ 代表 f 的顶点。用 $L(f)$ 代表 $L(top(f))$ 。用 $f=(n,b,c)$ 表示 $top(f)=n$ 且 $E(n)=\langle top(b),top(c)\rangle$ 。 设 f 为 OBDD 且其上变量的序为由小（根节点）到大（叶节点）。 则 $f|_{x=0}$ 和 $f|_{x=1}$ 的算法如下：

若 f 是叶节点，则 $f\|_{x=0}=f$
若 $L(f)=x$ 且 $f=(n,s_0,s_1)$ ，则 $f\|_{x=0}=s_0$
若 $L(f)\neq x$ 且 $f=(n,s_0,s_1)$ ，则 $f\|_{x=0}=(n,s_0\|_{x=0},s_1\|_{x=0})$
若 f 是叶节点，则 $f\|_{x=1}=f$
若 $L(f)=x$ 且 $f=(n,s_0,s_1)$ ，则 $f\|_{x=1}=s_1$
若 $L(f)\neq x$ 且 $f=(n,s_0,s_1)$ ，则 $f\|_{x=1}=(n,s_0\|_{x=1},s_1\|_{x=1})$

我们有 $\exists y_1...y_n.\exists x.f=\exists y_1...y_n.(f|_{x=0}\vee f|_{x=1})$ 。 OBDD 上的或运算以及其他运算定义如下。

OBDD 上的运算

若 $s=(n,s',s'')$ 且 $L(s)=x\in V$, 则将 s,s',s'' 分别表示为 $(\neg x\wedge s')\vee(x\wedge s''),s|_{x=0},s|_{x=1}$ 。若 $L(s)=0$ 或 $L(s)=1$ ，则用 0 或 1 直接表示该节点。

命题逻辑连接符可以看成是 OBDD 上的运算。设 $\circ$ 为二元运算符， f 和 g 为 OBDD 且其上变量的序为由小（根节点）到大（叶节点）。则 $f\circ g$ 的算法如下：

若 f 和 g 都是叶节点，则 $f\circ g=L(f)\circ L(g)$
若 $L(f)=L(g)=x$ 则 $f\circ g=((\neg x\wedge(f\|_{x=0}\circ g\|_{x=0})\vee(x\wedge(f\|_{x=1}\circ g\|_{x=1}))$
若 $x=L(f)<L(g)$ 则 $f\circ g=((\neg x\wedge(f\|_{x=0}\circ g)\vee(x\wedge(f\|_{x=1}\circ g))$
若 $L(f)>L(g)=y$ 则 $f\circ g=((\neg y\wedge(f\circ g\|_{y=0})\vee(y\wedge(f\circ g\|_{y=1}))$

常用的二元运算有 $\wedge,\vee,\to,\leftrightarrow$ 。一元运算 $\neg$ 原则上可用 $\to$ 实现，即 $\neg p\Leftrightarrow(p\to\text{false})$ ，其计算结果就是将叶结点的 0 和 1 的值互换。公式 φ_0 和 φ_1 是否等价可以根据 $\varphi_0\leftrightarrow\varphi_1$ 的 ROBDD 是否为 1 来判断。

量词消去的优化策略

用 $\bar{x}$ 表示 $x_1,...,x_n$ 。则 $ex(\varphi)=\exists x'_1...x'_n.(R(\bar{x},\bar{x'})\wedge\varphi^{\bar{x'}}_{\bar{x}})$ 。

计算 $ex(\varphi)$ 的 ROBDD 需要进行量词消去，是符号模型检测中计算量很大的操作。我们应用等价变换，尽可能将存在量词作用于较小的子公式。 迁移关系 $R(\bar{x},\bar{x'})$ 可能是一组公式的析取或合取。以下分别对这两种请况进行分析。

析取关系

设 $R(\bar{x},\bar{x'})=R_1(\bar{x},\bar{x'})\vee\cdots\vee R_m(\bar{x},\bar{x'})$ 。

我们应用等式 $\exists x.(p(x)\vee q(x))=\exists x.p(x)\vee\exists x.q(x))$ 来简化 ROBDD 的计算。我们有

$$\exists x'_1...x'_n.(R(\bar{x},\bar{x'})\wedge\varphi^{x'_1,...,x'_n}_{x_1,...,x_n})=\exists x'_1...x'_n.(R_1(\bar{x},\bar{x'})\wedge\varphi^{\bar{x'}}_{\bar{x}})\vee\cdots\vee\exists x'_1...x'_n.(R_n(\bar{x},\bar{x'})\wedge\varphi^{\bar{x'}}_{\bar{x}})$$

合取关系

设 $R(\bar{x},\bar{x'})=R_1(\bar{x},\bar{x'})\wedge\cdots\wedge R_m(\bar{x},\bar{x'})$ 。

若 x 不在 $p(x)$ 中出现，我们有 $\exists x.(p(x)\wedge q(x))=p(x)\wedge\exists x.q(x))$ 。我们应用该等式来简化 ROBDD 的计算。为方便起见，设 $R_0(\bar{x},\bar{x'})=\varphi^{x'_1,...,x'_n}_{x_1,...,x_n}$ 。则

$$\exists x'_1...x'_n.(R(\bar{x},\bar{x'})\wedge\varphi^{x'_1,...,x'_n}_{x_1,...,x_n})=\exists x'_1...x'_n.(R_0(\bar{x},\bar{x'})\wedge\cdots\wedge R_m(\bar{x},\bar{x'})$$

我们将 $x'_1, ..., x'_n$ 分成 k 个不相交子集 $d_1, ..., d_k$ ，将 $R_0, ..., R_m$ 分成 k 个不相交非空子集 $D_1, ..., D_k$ 。如果对于所有 $1 \leq i < j \leq k$ ， d_j 中的变量不在 D_i 中出现，则

$$\exists x'_1 ... x'_n.(R_0(\bar{x}, \bar{x'}) \wedge \cdots \wedge R_m(\bar{x}, \bar{x'}) = \exists d_1.(D_1 \wedge \exists d_2.(D_2 \wedge \cdots \wedge \exists d_k.(D_k) \cdots))$$

其中 d_i 表示 d_i 中变量的列举， D_i 表示 D_i 中公式的合取。对于给定的公式集合 $\{R_0, ..., R_m\}$ 和变量集合 $\{x'_1, ..., x'_n\}$ ，我们可以依据不同的标准来计算 $d_1, ..., d_k$ 和 $D_1, ..., D_k$ 。原则上我们应该有外层的 d_i 越小而 D_i 越大越好。

§5.2 基于路径分析的模型检测

基于路径的分析的出发点是确定模型的每条路径是否满足模些性质。基于状态分析的模型检测适用于 PLTL 性质。一个系统可以看作是其可能运行的路径的集合。一个系统满足某个路径公式，就是说系统的所有运行都满足这个公式。 设 $M = \langle S, \Delta, I, L \rangle$ 是 AP 上的标号 Kripke 结构， φ 是 AP 上的 PLTL 公式。

$$M \models \varphi \text{ 当且仅当 } [[M]] \subseteq [[\varphi]] \text{ 。}$$

§5.2.1 Kripke 结构与 Büchi 自动机

设 $M = \langle S, \Delta, I, L \rangle$ 是 AP 上的标号 Kripke 结构。定义

$$\begin{array}{|l|}\hline \Sigma = 2^{AP} \\ \Delta' = \{(s, a, s') \mid (s, s') \in \Delta, a = L(s)\} \\ F = S \\ \hline \end{array}$$

令 $A_M = (\Sigma, S, \Delta', I, F)$ 。则 $[[A_M]] = [[M]$ 。

称 A_M 为 AP 上的 Kripke 结构 $M = \langle S, \Delta, I, L \rangle$ 所对应的 Büchi 自动机。

§5.2.2 模型检测原理

由于 Kripke 结构和 PLTL 公式都可以转换成对应的 Büchi 自动机。模型检测问题可转化为两个自动机的语言的包含关系或一个自动机的语言是否为空的问题。用 A_φ 表示公式 φ 对应的 Büchi 自动机。 一个给定的系统 M 满足一个 AP 上的时序逻辑公式 φ ，即：

$$M \models \varphi$$

即 $[[M]] \subseteq [[\varphi]]$

即 $[[M]] \cap ((2^{AP})^\omega \setminus [[\varphi]]) = \emptyset$

即 $[[M]] \cap [[\neg\varphi]] = \emptyset$

即 $[[A_M]] \cap [[A_{\neg\varphi}]] = \emptyset$

即 $A_M \cap A_{\neg\varphi}$ 的语言为空。模型检测问题转化为自动机非空问题。

双深度优先搜索算法

为简单起见，我们使用栈作为算法的基本数据结构。栈的三个基本操作为压栈、退栈和读取栈最上面的一个元素。设 W 为一个栈， q 为一个元素。这三个操作分别记作 push(q,W) 、 pop(W) 、 top(W) 。记空栈为 [ ] 。 给定 Büchi 自动机 $\langle \Sigma, S, \Delta, I, F \rangle$ ，其语言是否为空可以用以下算法计算。定义 $\Delta(q) = \{s \mid \exists a \in \Sigma.(q, a, s) \in \Delta\}$ 为 q 的所有后继状态的集合。该算法由两个嵌套的深度优先搜索算法实现。

```
start()
{
  W = A = B = [ ];
  for each s ∈ I, if (s ∉ A) { push(s,A); push(s,W); dfs1(s); pop(W); }
  report("empty");
}
```

```
dfs1(q)
{
  for each s ∈ Δ(q), if (s ∉ A) { push(s,A); push(s,W); dfs1(s); pop(W); }
  if (q ∈ F) { push(s,B); dfs2(s); }
}
```

```
dfs2(q)
{
  for each s ∈ Δ(q), { if (s ∈ W) report("nonemp"); if (s ∉ B) { push(s,B); dfs2(s); } }
}
```

给定一个 Büchi 自动机，算法报告 "nonemp" 当且仅当该自动机的语言非空。该判定算法的状态搜索次数相对于系统规模是线性的。

动态模型检测原理

给定一个隐式迁移系统描述和一个 PLTL 性质描述。由于计算效率的问题，验证这样的一个问题并不先构造系统的 Kripke 结构。计算 $A_M \cap A_{\neg\varphi}$ 是否为空的算法可以在 $A_{\neg\varphi}$ 已经存在的情况下，边构造 A_M 边计算 $A_M \cap A_{\neg\varphi}$ 。一条有利的因素是在根据系统模型的高层描述构造新的系统状态时，可以判断这些新的状态是否与 $A_{\neg\varphi}$ 有共同部分，决定是否需要对这些状态进一步扩展。另外就是我们只要判定 $A_M \cap A_{\neg\varphi}$ 是否为空，因此找到一条满足 $A_M \cap A_{\neg\varphi}$ 的运行就可退出，不必一定等到完成 A_M 的构造。

踏步等价与偏序归约

两条无穷路径 π 和 π' 是踏步等价的当前仅当存在两条整数上的无穷序列 $0 = i_0 < i_1 < i_2 < \cdots$ 和 $0 = j_0 < j_1 < j_2 < \cdots$ 使得对任意 $k \geq 0$,

$$L(\pi_{i_k}) = L(\pi_{i_k+1}) = \cdots = L(\pi_{i_{k+1}-1}) = L(\pi'_{j_k}) = L(\pi'_{j_k+1}) = \cdots = L(\pi'_{j_{k+1}-1})$$

一个公式 φ 是踏步不变的当前仅当对任意踏步等价的 π 和 π' ,

$$\pi \models \varphi \Leftrightarrow \pi' \models \varphi$$

不出现 O 算子的 PLTL 公式的集合记为 $PLTL_{-X}$ 。该集合的公式都是踏步不变的。

设要验证的性质为 $\varphi \in PLTL_{-X}$ 且 φ 中的命题变量的集合为 AP' 。假设系统的运行集合包含 $\pi = s_0 ... s_i s_{i+1} s_{i+2} ...$ 和 $\pi' = s_0 ... s_i s'_{i+1} s_{i+2} ...$ 。

如果 $L(s_i) \cap AP' = L(s_{i+1}) \cap AP'$ 且 $L(s'_{i+1}) \cap AP' = L(s_{i+2}) \cap AP'$, 则 $\pi \models \varphi$ 当且仅当 $\pi' \models \varphi$ 。因此对于验证系统是否满足 φ , 我们可以不考虑 π' 。

偏序归约的主要思想就是根据一些判断条件，在动态模型检测的使用中避免类似于 s'_{i+1} 这样的状态的生成以及对 π' 的检查。

§5.3 限界模型检测

限界模型检测的主要思路是对参与检测的路径做限制，如果在参与检测的路径长度较短时能知道一个公式是否满足，则该方法有较好的效率。

这一节所有的公式都必须是 NNF 范式。若出现的公式不是 NNF 范式，则将其看成是与其等价的公式的 NNF 范式。

§5.3.1 CTL 公式的限界模型检测

给定 AP 上的 Kripke 结构 $M = \langle S, R, I, L\rangle$ 。M 的 k 界模型, 简称 k 模型 $M_k = \langle S, P_k, I, L\rangle$ 其中 P_k 为 M 中长度为 $k+1$ 的路径的集合。设 $\pi \in P_k$ 。这样的路径称为 k 路径。定义

$$rs(\pi) = \bigvee_{i=0}^{|\pi|-2} \bigvee_{j=i+1}^{|\pi|-1} \pi_i = \pi_j$$

CTL 公式的限界语义

NNF 范式的 CTL 公式 φ 的限界语义 $M_k, s \models \varphi$ 定义如下：

$M_k, s \models p$	若 $p = \top$, 或 $p \in AP$ 且 $p \in L(s)$
$M_k, s \models \neg p$	若 $M_k, s \not\models p$
$M_k, s \models \varphi \vee \psi$	若 $M_k, s \models \varphi$ 或 $M_k, s \models \psi$
$M_k, s \models \varphi \wedge \psi$	若 $M_k, s \models \varphi$ 且 $M_k, s \models \psi$
$M_k, s \models A\varphi$	若对于所有 P_k 中以 s 为起点的路径 π ： $M_k, \pi \models \varphi$
$M_k, s \models E\varphi$	若存在 P_k 中以 s 为起点的路径 π ： $M_k, \pi \models \varphi$
$M_k, \pi \models X\varphi$	若 $k \geq 1$ 且 $M_k, \pi_1 \models \varphi$
$M_k, \pi \models G\psi$	若 $rs(\pi)$ 且 $\forall i \leq k, M_k, \pi_i \models \psi$
$M_k, \pi \models F\varphi$	若 $\exists i \leq k, M_k, \pi_i \models \varphi$
$M_k, \pi \models \varphi U \psi$	若 $\exists i \leq k, M_k, \pi_i \models \psi$ 且 $\forall j < i, M_k, \pi_j \models \varphi$
$M_k, \pi \models \varphi R \psi$	若 $\forall i \leq k$, 若 $\forall j < i, M_k, \pi_j \not\models \varphi$ 则 $M_k, \pi_i \models \psi$, 且若 $\forall i \leq k, M_k, \pi_i \not\models \varphi$ 则 $rs(\pi)$

限界语义的正确性和完备性

我们有

$$M, s \models \varphi \text{ 当且仅当存在 } k \geq 0 \text{ 使得 } M_k, s \models \varphi \text{ 。}$$

定义 $M_k \models \varphi$ 当且仅当

$$\text{对所有 } s \in I \text{ ，} \quad M_k, s \models \varphi \text{ 。}$$

我们有

$$M \models \varphi \text{ 当且仅当存在 } k \geq 0 \text{ 使得 } M_k \models \varphi \text{ 。}$$

限界模型检测算法

给定模型 M 和公式 φ 。限界模型检测算法如下：

(1) $k = 0$
(2) 若 $M_k \models \varphi$ 成立，则 $M \models \varphi$ ，算法结束
(3) 若存在 s 使得 $M_k, s \models \neg\varphi$ 成立，则 $M \not\models \varphi$ ，算法结束
(4) $k = k + 1$ ，回到 (2)

注意：这里的 $\neg\varphi$ 代表与其等价的 CTL 公式的 NNF 范式。前面的性质保证了算法一定能够终止和获得正确答案。

§5.3.2 PLTL 公式的限界模型检测

对于 PLTL，我们的关注是否存在一个运行满足给定的公式。对于 k 路径，我们需要考虑这条路径的终点是否有可能回到已经经过的点形成一个环。这样的环隐含一条无穷路径，可以当成无穷路径对待。定义

$$s(i,k,n)=\begin{cases} n & \text{若 } i=k \text{ 。} \\ i+1 & \text{若 } i<k \text{ 。} \end{cases}$$

PLTL 公式的限界语义

定义 $M_k,\pi\models_n^i\varphi$ 如下：

$M_k,\pi\models_n^i p$	若 $p=\top$, 或 $p\in AP$ 且 $p\in L(\pi_i)$
$M_k,\pi\models_n^i \neg p$	若 $M_k,\pi\not\models_n^i p$
$M_k,\pi\models_n^i \varphi\wedge\psi$	若 $M_k,\pi\models_n^i\varphi$ 且 $M_k,\pi\models_n^i\psi$
$M_k,\pi\models_n^i \varphi\vee\psi$	若 $M_k,\pi\models_n^i\varphi$ 或 $M_k,\pi\models_n^i\psi$
$M_k,\pi\models_n^i X\varphi$	若 $M_k,\pi\models_n^{s(i,k,n)}\varphi$
$M_k,\pi\models_n^i G\varphi$	若 $\forall j\in\{\min(i,n),...,k\}, M_k,\pi\models_n^j\varphi$
$M_k,\pi\models_n^i \varphi U\psi$	若 $\exists j\in\{\min(i,n),...,k\}$, $M_k,\pi\models_n^j\psi$ 且 $i\le j\to\forall m\in\{i,...,j-1\}, M_k,\pi\models_n^m\varphi$ 且 $i>j\to\forall m\in\{n,...,j-1,i,...,k\}, M_k,\pi\models_n^m\varphi$

定义 $M_k,\pi\models_n^{-1}\varphi$ 为永假。定义 $R(\pi_k,\pi_{-1})$ 为永真。
NNF 范式的 LTL 公式 φ 的限界语义 $M_k,\pi\models\varphi$ 定义如下：

$M_k,\pi\models\varphi$ 当且仅当存在 $n\in\{-1,0,...,k\}$，使得 $R(\pi_k,\pi_n)$ 且 $M_k,\pi\models_n^0\varphi$ 。

限界语义的正确性和完备性

我们有

$\exists\pi\in M,\pi_0\in I.(M,\pi\models\varphi)$ 当且仅当存在 $k\ge 0$，$\exists\pi\in M_k,\pi_0\in I.(M_k,\pi\models\varphi)$ 。

定义

$M_k\models\varphi$ 当且仅当不存在始于 I 的路径 $\pi\in P_k$ 满足 $M_k,\pi\models\neg\varphi$ 。

我们有

$M\models\varphi$ 当且仅当 $\forall k\ge 0, M_k\models\varphi$ 。

限界模型检测算法

我们有 $M_k\not\models\varphi$ 则对所有 $i\ge 0, M_{k+i}\not\models\varphi$ 。定义 (M,φ) 的完备阈值 $ct(M,\varphi)$ 为最小的满足以上条件的 k 。对任何 M 和 φ，完备阈值 $ct(M,\varphi)$ 存在。进而我们有

$M\models\varphi$ 当且仅当对 $k=ct(M,\varphi), M_k\models\varphi$ 。

给定模型 M，φ 和 $m\ge ct(M,\varphi)$ 。限界模型检测算法如下：

(1) $k=0$
(2) 若 $M_k \not\models \varphi$，则 $M \not\models \varphi$，算法结束
(3) 若 $k=m$，则 $M \models \varphi$，算法结束
(4) $k=k+1$，回到 (2)

计算 $M_k \not\models \varphi$ 即根据限界语义计算是否存在初始状态出发的 k 路径 π 使得 $M_k, \pi \models \neg\varphi$。假定我们通过估算能够正确地获得 $m \geq ct(M,\varphi)$，前面的性质保证了算法一定能够终止和获得正确答案。

§5.4 公平性

本章前面提到的系统模型以 Kripke 结构为基础，并不能体现公平性约束。一般而言，公平性可以用状态集合或用状态集合的二元组表示，而状态集合可用公式表示。对有些特殊的公平性可以用特殊的申明表示。

关于给定公式集合的弱公平性

给定公式集合 $\{\varphi_1, ..., \varphi_n\}$。一个运行满足弱公平性当且仅当对每个 i，该运行无限次满足 φ_i。

关于给定公式二元组集合的弱公平性

给定公式二元组集合 $\{(\varphi_1,\psi_1), ..., (\varphi_n,\psi_n)\}$。一个运行满足弱公平性当且仅当对每个 i，该运行如果从某状态起总是满足 φ_i，则在该状态后有状态满足 ψ_i。这个弱公平性等价于关于公式集合 $\{(\neg\varphi_1 \vee \psi_1), ..., (\neg\varphi_n \vee \psi_n)\}$ 的弱公平性。

关于给定公式二元组集合的强公平性

给定公式二元组集合 $\{(\varphi_1,\psi_1), ..., (\varphi_n,\psi_n)\}$。一个运行满足强公平性当且仅当对每个 i，该运行如果从某状态起无限次地满足 φ_i，则在该状态后有状态满足 ψ_i。

关于进程和迁移的公平性

假设系统由若干进程组成，进程由若干迁移组成。我们可以申明

进程弱公平性：	若某进程总是可执行的，则该进程必须执行一次。
迁移弱公平性：	若某迁移总是可执行的，则该迁移必须执行一次。
进程强公平性：	若某进程总有可执行的时候，则该进程必须执行一次。
迁移强公平性：	若某迁移总有可执行的时候，则该迁移必须执行一次。

公平系统的模型检测

以上所述公平性在系统模型中具有一定的普遍性，很多系统模型描述方法与模型检测工具提供描述一种或多种以上所述公平性的方法，并实现相关模型检测算法。

§5.5 模型检测工具

基于 OBDD 的状态分析的模型检测的工具有 SMV 和 NuSMV 等。 SMV 最早由 CMU 大学开发，主要应用领域包括电子电路验证等方面。 基于路径分析的模型检测工具有 SPIN 等。 SPIN 由贝尔实验室开发，主要应用领域包括通信协议验证等方面。 基于限界语义的模型检测工具有 VERDS 等。 VERDS 由中国科学院软件研究所计算机科学国家重点实验室开发，是有穷状态并发迁移系统的符号模型检测与限界模型检测工具。

§5.6 说明

本章介绍模型检测方法和工具。符号模型检测和模型检测工具 SMV 可参考 [McM93] 。基于自动机原理的模型检测和模型检测工具 SPIN 可参考 [Hol90, Hol03] 。基于限界语义的模型检测和模型检测工具 VERDS 可参考 [BCCZ99, Zha07, Zha09] 。

参考文献

[BCCZ99] A. Biere, A. Cimmatti, E. Clarke, and Y. Zhu. Symbolic Model Checking without BDDs. LNCS 1579:193-207. TACAS 99.

[McM93] K. L. McMillan. Symbolic Model Checking. Kluwer Publisher, 1993.

[Hol90] G. J. Holzmann. Design and Validation of Computer Protocols. New Jersey: Prentice Hall, 1990.

[Hol03] G. J. Holzmann. The SPIN Model Checker. Boston: Addison-Wesley, 2003.

[Zha07] W. Zhang. Model Checking with SAT-Based Characterization of ACTL Formulas. Lecture Notes in Computer Science 4789 (ICFEM 2007):191-211. Springer-Verlag. 2007.

[Zha09] W. Zhang. Bounded Semantics of CTL and SAT-based Verification. Lecture Notes in Computer Science 5885 (ICFEM 2009):286-305. Springer-Verlag. 2009.