

中国科学院大学课程讲义

形式化方法

中国科学院大学计算机科学与技术学院

2021年5月31日

前言

形式化方法的目标是开发满足给定需求的程序与软件，即具有正确性保证的程序与软件。形式化方法依赖的理论与技术是软件系统行为的数学模型、逻辑描述、以及形式化的分析验证方法。

程序与软件的正确性问题就是在程序与软件基础上建立的计算机软件系统的行为是否合乎行为规范的问题。这里涉及三个方面：软件系统行为、行为规范、软件系统行为是否符合行为规范的分析验证。软件系统行为可抽象地用具有状态变化的动态系统来描述。一类描述这类系统的数学模型称为迁移系统。本书主要介绍形式化方法的基本概念和迁移系统及其行为规范的描述方法与分析验证方法，以离散迁移系统为主。相关行为规范的描述语言包括用于描述顺序计算系统性质的一阶逻辑和描述并发系统性质的时序逻辑。相关分析验证方法包括基于推理规则的演绎推理和基于状态搜索的模型检测。

本书内容的章节划分如下。第1章介绍预备知识，第2-4章介绍不同类型的迁移系统，第5-6章介绍时序逻辑，第7-8章分别介绍演绎推理和模型检测方法，第9章介绍实例分析。附录A介绍第9章的实例分析中用到的模块化迁移系统建模语言。附录B为参考文献。若本书应用于研究生教学，由于学生的基础知识可能存在一定的差别，预备知识部分可先略过，其后章节内容中若有用到学生可能缺乏的相关预备知识时再稍加回顾；实例分析的部分内容可在介绍理论与方法时一并介绍且可进行适当简化裁剪以服务于这些方法的应用举例并对不同方法在同一个验证问题上的应用进行对照比较。

形式化方法虽然在理论上具有保证程序与软件在模型层次上的正确性的优越性，但对于实际应用而言，由于形式描述和验证的复杂性，其使用范围十分受限。当前众多的相关研究正试图为形式化方法理论和应用中的一些问题提供解决方案，本书涉及基本的概念和方法，希望其内容能够帮助读者在形式化方法的研究和应用方面提供相关基础知识和基本方法。

目 录

1	预备知识	1
1.1	命题逻辑	1
1.2	谓词逻辑	2
1.3	集合	4
1.4	关系	5
1.5	函数	7
1.6	完全偏序和格上的不动点	8
1.7	有向图	9
1.8	练习	10
2	基于状态变迁的迁移系统	11
2.1	Kripke模型	11
2.1.1	可达性质分析	12
2.1.2	安全性质分析	13
2.1.3	可避免性质分析	14
2.1.4	必达性质分析	16
2.2	公平Kripke模型	17
2.2.1	公平可达性质分析	19
2.2.2	公平安全性质分析	20
2.2.3	公平可避免性质分析	20
2.2.4	公平必达性质分析	21
2.2.5	模型非空问题	22
2.2.6	非空问题的应用	22
2.2.7	强公平与弱公平条件	24
2.3	标号Kripke模型	25
2.4	公平标号Kripke模型	27
2.5	例子	28
2.6	练习	33
3	基于一阶逻辑的迁移系统	34
3.1	卫式迁移模型	34
3.1.1	正确性性质	35
3.1.2	卫式迁移模型与标号Kripke模型的等价	36
3.2	公平卫式迁移模型	37
3.2.1	正确性性质	37
3.2.2	公平卫式迁移模型与公平标号Kripke模型的等价	37
3.3	谓词公式迁移模型	38
3.4	流程图模型	39
3.4.1	正确性问题	41

3.4.2	流程图模型与卫式迁移模型的等价	42
3.4.3	并发流程图模型	43
3.5	结构化程序模型	44
3.5.1	正确性问题	45
3.5.2	结构化程序模型与流程图模型的等价	46
3.6	例子	46
3.7	练习	50
4	基于迁移标号的迁移系统	51
4.1	标号迁移系统	51
4.2	无穷字符串上的自动机	51
4.2.1	自动机的可接受运行条件的设置	53
4.2.2	确定型自动机与非确定型自动机	55
4.2.3	自动机与Kripke模型	55
4.3	时间迁移系统与时间自动机	56
4.4	混成迁移系统	57
4.5	Petri网模型	58
4.6	通信系统	60
4.6.1	通道	60
4.6.2	通信单元	61
4.6.3	通信系统	61
4.6.4	通信单元与卫式迁移模型的等价	62
4.7	例子	63
4.8	练习	69
5	线性时序逻辑	71
5.1	命题线性时序逻辑(PLTL)	71
5.1.1	PLTL公式的推理	73
5.1.2	PLTL限界语义	74
5.1.3	PLTL公式的Büchi自动机表示	75
5.2	PLTL公式的不动点表示与线性 μ 演算(ν TL)	76
5.2.1	线性 μ 演算:	77
5.2.2	PLTL公式到 ν TL公式的转换:	78
5.2.3	公平标号Kripke结构上的LTL语义	78
5.3	一阶线性时序逻辑	79
5.4	例子	81
5.5	练习	87
6	分枝时序逻辑	88
6.1	计算树逻辑CTL	88
6.1.1	CTL公式的推理	89

6.1.2	CTL限界语义	90
6.2	CTL公式的不动点表示与模态 μ 演算	91
6.2.1	模态 μ 演算	92
6.2.2	CTL公式到 μ 演算公式的转换	93
6.2.3	公平标号Kripke结构下的CTL语义	93
6.3	计算树逻辑CTL*	94
6.4	例子	95
6.5	练习	98
7	基于演绎推理的验证方法	99
7.1	卫式迁移模型的推理	99
7.2	流程图模型的推理	101
7.2.1	直接证明	101
7.2.2	基于路径的推理	102
7.3	结构化程序模型的推理	104
7.3.1	指称语义	104
7.3.2	Hoare逻辑	105
7.3.3	Hoare逻辑的扩展	109
7.4	例子	110
7.5	练习	120
8	基于模型检测的验证方法	122
8.1	符号模型	122
8.2	CTL性质的模型检测	124
8.2.1	CTL性质的状态标号算法与不动点算法	124
8.2.2	CTL性质的符号模型检测	125
8.2.3	CTL性质的限界正确性检查	126
8.3	PLTL性质的模型检测	127
8.3.1	PLTL性质的基于自动机空性检测的算法	127
8.3.2	PLTL性质的符号模型检测	127
8.3.3	PLTL公式的限界模型检测	127
8.4	公平标号Kripke结构的模型检测	128
8.5	模型检测技术	129
8.5.1	二元决策图	129
8.5.2	动态模型检测	132
8.6	例子	132
8.7	练习	133
9	实例分析	135
9.1	互斥算法	135
9.1.1	演绎推理	136

9.1.2	基于模型检测算法的验证	138
9.1.3	基于模型检测工具的自动验证	140
9.1.4	公平约束模型的模型检测	140
9.2	窗口协议	141
9.2.1	可靠通道的窗口协议的卫式迁移模型:	143
9.2.2	不可靠通道的窗口协议的卫式迁移模型:	146
9.2.3	窗口协议的客户模型	147
9.3	整数平方根算法	148
9.3.1	演绎推理	149
9.3.2	基于模型检测工具的自动验证	151
9.3.3	错误检查与模型检测中的反例生成	151
9.4	模型中特定状态和路径的查找	152
9.5	练习	153
A	建模语言VML	154
B	参考文献	162

§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， $\varphi_1, \dots, \varphi_n$ 是由 S 生成的公式，则 $f(\varphi_1, \dots, \varphi_n)$ 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 $\varphi = f(\varphi_1, \dots, \varphi_n)$ ，其中 f 是 S 中的 n 元($n \geq 1$)联结词，则 $v(\varphi) = f(v(\varphi_1), \dots, v(\varphi_n))$ 。

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, \dots, A_n 的赋值为 a_1, \dots, a_n 得到的赋值记作 $v[A_1/a_1, \dots, A_n/a_n]$ 。设 $v' = v[A_1/a_1, \dots, A_n/a_n]$ 。我们有

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

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

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

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, \dots, A_n 是不同的命题变元。如果公式 φ 中不出现除 A_1, \dots, A_n 之外的命题变元，且 $\varphi = f(A_1, \dots, A_n)$ ，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 ν 满足 Γ 中的每个公式，则称 ν 满足 Γ 。如果有真值赋值 ν 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 $\Gamma \vdash \varphi$ 。 $\Gamma \vdash \varphi$ 不成立记作 $\Gamma \not\vdash \varphi$ 。若 $\Gamma = \{\varphi_1, \dots, \varphi_n\}$ ，则将 $\Gamma \vdash \varphi$ 写作 $\varphi_1, \dots, \varphi_n \vdash \varphi$ 。

设 $\varphi_1, \dots, \varphi_n, \varphi, \psi$ 为公式。 $\vdash \varphi$ 当且仅当 φ 是永真式。 $\varphi_1, \dots, \varphi_n \vdash \varphi$ 当且仅当 $\varphi_1 \wedge \dots \wedge \varphi_n \rightarrow \varphi$ 是永真式。 $\varphi \leftrightarrow \psi$ 当且仅当 $\varphi \vdash \psi$ 且 $\psi \vdash \varphi$ 。 $\Gamma \cup \{\varphi\} \vdash \psi$ 当且仅当 $\Gamma \vdash \varphi \rightarrow \psi$ 。 $\Gamma = \{\varphi_1, \dots, \varphi_n\}$ 是可满足的当且仅当 $\varphi_1 \wedge \dots \wedge \varphi_n$ 是可满足的。设 Γ 是公式集合。则 Γ 是不可满足的当且仅当每个公式都是 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, \dots, t_n 是由 F 生成的项，则 $f(t_1, \dots, t_n)$ 是由 F 生成的项。

设 G 是谓词符号的集合。若 t_1, \dots, t_n 是由 F 生成的项， A 是 P 中的 n 元谓词符号，则 $A(t_1, \dots, t_n)$ 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， $\varphi_1, \dots, \varphi_n$ 是 \mathcal{L}^B 的公式，则 $f(\varphi_1, \dots, \varphi_n)$ 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, \dots, x_n 的赋值为 a_1, \dots, a_n 得到的赋值记作 $\sigma[x_1/a_1, \dots, x_n/a_n]$ 。我们有

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

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 $f(t_1, \dots, t_n)$, 其中 f 是 n 元函数符号, t_1, \dots, t_n 是项, 则 $I(t)\sigma = I_0(f)(I(t_1)\sigma, \dots, I(t_n)\sigma)$ 。

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 $P(t_1, \dots, t_n)$, 其中 P 是 n 元谓词符号, t_1, \dots, t_n 是项, 则 $I(\varphi)\sigma = I_0(P)(I(t_1)\sigma, \dots, I(t_n)\sigma)$; (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 $\varphi_0 \rightarrow \varphi_1$, 则 $I(\varphi)\sigma = I(\varphi_0)\sigma \rightarrow I(\varphi_1)\sigma$; (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 $Var(\varphi) = \{x_1, \dots, x_n\}$, 则称公式 $\forall x_1 \dots \forall x_n \varphi$ 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, \dots, x_n 是不同的变元, t_1, \dots, t_n 是项, 则称 $\{x_1/t_1, \dots, x_n/t_n\}$ 为代换。若 t 是项, 则 $t\{x_1/t_1, \dots, x_n/t_n\}$ 是用 t_1, \dots, t_n 分别替换 t 中 x_1, \dots, x_n 的所有出现得到的项, 记为 $t_{x_1, \dots, x_n}^{t_1, \dots, t_n}$ 。若 φ 是公式, 则 $\varphi\{x_1/t_1, \dots, x_n/t_n\}$ 是用 t_1, \dots, t_n 分别替换 φ 中 x_1, \dots, x_n 的所有自由出现得到的公式, 记为 $\varphi_{x_1, \dots, x_n}^{t_1, \dots, t_n}$ 。如果在公式 φ 和 $\varphi_{x_1, \dots, x_n}^{t_1, \dots, t_n}$ 中变元的约束出现次数相同, 则称 t_1, \dots, t_n 对于 φ 中的 x_1, \dots, x_n 是可代入的。若 t_1, \dots, t_n 对于 φ 中的 x_1, \dots, x_n 是可代入的, 则有

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

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 $\varphi_1, \dots, \varphi_i$ 分别替换命题逻辑公式 φ 中的命题变元 A_1, \dots, A_n 得到的谓词逻辑公式记为 $\varphi_{A_1, \dots, A_n}^{\varphi_1, \dots, \varphi_n}$, 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 $\Gamma = \{\varphi_1, \dots, \varphi_n\}$, 则将 $\Gamma \models \varphi$ 写作 $\varphi_1, \dots, \varphi_n \models \varphi$ 。

设 $\varphi_1, \dots, \varphi_n, \varphi, \psi$ 为公式。 $\models \varphi$ 当且仅当 φ 是永真式。 $\varphi_1, \dots, \varphi_n \models \varphi$ 当且仅当 $\varphi_1 \wedge \dots \wedge \varphi_n \rightarrow \varphi$ 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 $\Gamma \models \varphi \rightarrow \psi$ 。 $\Gamma = \{\varphi_1, \dots, \varphi_n\}$ 是可满足的当且仅当 $\varphi_1 \wedge \dots \wedge \varphi_n$ 是可满足的。设 Γ 是公式集合。则 Γ 是不可满足的当且仅当每个公式都是 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。

$$\begin{aligned} (A = B) &\Leftrightarrow \forall x(x \in A \leftrightarrow x \in B) \\ (A \subseteq B) &\Leftrightarrow \forall x(x \in A \rightarrow x \in B) \end{aligned}$$

子集关系满足以下性质。

$$\begin{aligned} A &\subseteq A \\ A \subseteq B \text{ 且 } B \subseteq C &\text{ 则 } A \subseteq C \\ A \subseteq B \text{ 且 } B \subseteq A &\text{ 则 } A = B \end{aligned}$$

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 2^A , 即 $2^A = \{X \mid X \subseteq A\}$ 。若 $a \in A$, 则 $\{a\} \subseteq A$ 。若 $A \subseteq B$, 则 $A \in 2^B$ 。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 $|2^A| = 2^{|A|}$ 。

集合的运算有交、并、差。

$$\begin{aligned} \text{交} \quad A \cap B &= \{x \mid x \in A \wedge x \in B\} \\ \text{并} \quad A \cup B &= \{x \mid x \in A \vee x \in B\} \\ \text{差} \quad A - B &= \{x \mid x \in A \wedge x \notin B\} \end{aligned}$$

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 $\sim A$ 。

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。

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

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 $\langle x_1, \dots, x_{n-1}, x_n \rangle = \langle \langle x_1, \dots, x_{n-1} \rangle, x_n \rangle$ 。 $\langle x_1, \dots, x_{n-1}, x_n \rangle = \langle y_1, \dots, y_{n-1}, y_n \rangle$ 当且仅当 $x_1 = y_1, \dots, x_{n-1} = y_{n-1}$ 且 $x_n = y_n$ 。

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

§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 $\forall x_1 \dots \forall x_n (x_1 \in X \wedge \dots \wedge x_n \in X \rightarrow \varphi)$ 为 $\forall x_1, \dots, x_n \in X. \varphi$ 。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	$\forall x, y \in X. (xRy \rightarrow yRx)$
反对称性	$\forall x, y \in X. (xRy \wedge yRx \rightarrow x = y)$
传递性	$\forall x, y, z \in X. (xRy \wedge yRz \rightarrow xRz)$

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。 $R^0 = I_X = \{(x, x) \mid x \in X\}$ 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

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

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 $A = \{A_1, \dots, A_n\}$ 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。

满足自反性和传递性的非空集合上的关系称为预序关系。如果 \leq 是 X 上的预序关系，那么有序偶 $\langle P, \leq \rangle$ 表示预序集合。

给定预序集合 $\langle P, \leq \rangle$ 和 $Q \subseteq P$ 。如果所有 P 的不包含 Q 元素的非空子集都有极小元且若 $a \leq b \leq a$ 且 $a \neq b$ 则 $a, b \in Q$ ，则称 $\langle P, \leq \rangle$ 为 Q 弱基集合。一个预序集合是 Q 弱基集合当且仅当无限严格递减序列中非 Q 元素只出现有限多次。

设 $\langle P, \leq \rangle$ 为 Q 弱基集合。记 $x \leq y$ 且 $x \neq y$ 为 $x < y$ 。记 $\Delta(Q)(x)$ 为以 x 为出发点的只包含 Q 元素的无限严格递减序列的集合。以下为弱基归纳推理。

$$\text{若 } \forall x' \in P. (\forall x \in P. (x < x' \rightarrow (\Delta(Q)(x) = \emptyset \rightarrow \varphi(x))) \rightarrow \varphi(x')) \text{ 则 } \forall x \in P. \varphi(x).$$

给定预序集合 $\langle P, \leq \rangle$ 。若 $\langle P, \leq \rangle$ 是 \emptyset 弱基集合，则称 $\langle P, \leq \rangle$ 为良基(Well-Founded)集合。

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。

若 $\langle P, \leq \rangle$ 是良基集合，则 $\langle P, \leq \rangle$ 是偏序集合。

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。

设 $\langle P_1, \leq_1 \rangle, \dots, \langle P_n, \leq_n \rangle$ 为偏序, $P = P_1 \times \dots \times P_n$ 。则偏序 $\langle P_1, \leq_1 \rangle, \dots, \langle P_n, \leq_n \rangle$ 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 $i \in \{1, \dots, n\}$, 有 $pr_i(a) \leq_i pr_i(b)$ 。设 $S \subseteq P_1 \times \dots \times P_n$ 。则 $\sqcup S$ 存在当且仅当对于所有 $i \in \{1, \dots, n\}$, $\sqcup pr_i(S)$ 存在, 且 $\sqcup S = \langle \sqcup pr_1(S), \dots, \sqcup pr_n(S) \rangle$ 。

§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 $f: X \rightarrow Y$ 。 X 称为 f 的定义域, Y 称为 f 的值域。

对于函数 $f: X \rightarrow Y$, 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 $f: X \rightarrow Y$ 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 $f: X' \rightarrow Y$ 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 $f: X \rightarrow Y$ 和 $g: Y \rightarrow Z$ 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 $f: X \rightarrow X$ 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。 $f^0(x) = I_X(x)$; $f^{m+1}(x) = f(f^m(x))$ 。

记 $\exists x_1 \dots \exists x_n (x_1 \in X \wedge \dots \wedge x_n \in X \wedge \varphi)$ 为 $\exists x_1, \dots, x_n \in X. \varphi$ 。设 $f: X \rightarrow Y$ 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	$\forall x_1, x_2 \in X. (f(x_1) = f(x_2) \rightarrow x_1 = x_2)$
f 是双射的	f 是满射的且是入射的

若 $f: X \rightarrow Y, g: Y \rightarrow Z$ 都是满射函数, 则 $g \circ f$ 也是满射函数; 若 $f: X \rightarrow Y, g: Y \rightarrow Z$ 都是入射函数, 则 $g \circ f$ 也是入射函数; 若 $f: X \rightarrow Y, g: Y \rightarrow Z$ 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 $f : X \rightarrow Y$ 是双射的，则其反函数 $f^{-1} : Y \rightarrow X$ 也是双射的。若 $f : X \rightarrow Y, g : Y \rightarrow Z$ 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 $\Psi_A : U \rightarrow \{0, 1\}$ 定义如下。

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

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 $(X \rightarrow Y)$ 。设 X 是一个集合， $\langle Y, \leq \rangle$ 是一个偏序， $f, g : X \rightarrow Y$ 是两个函数。 $f \leq g$ 当且仅当对于所有 $x \in X$ ， $f(x) \leq g(x)$ 。 $\langle (X \rightarrow Y), \leq \rangle$ 构成一个偏序。

设 $S \subseteq (X \rightarrow Y)$ 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 $x \in X$ ， $(\sqcup S)(x) = \sqcup S(x)$ 。

设 X 是一个集合。记 X 的大小为 n 的子集的集合为 $[X]^n$ 。一个 $[X]^n$ 的 k 染色函数 C 是一个 $[X]^n$ 到一个大小为 k 的集合的函数。称 H 为 C 的齐性集合当且仅当 C 作用在任意 $[H]^n$ 中的元素所得的值是一样的。以下结论称为Ramsey定理。

设 N 是自然数集合。任意 $[N]^n$ 的 k 染色函数都有一个无穷的齐性集合。

§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 \perp_X 或 \perp 。

任何有最小元且只包含有穷链的偏序是完全偏序。如果 $\langle P_1, \leq_1 \rangle, \dots, \langle P_n, \leq_n \rangle$ 是完全偏序，则其笛卡尔积 $\langle P_1 \times \dots \times P_n, \leq \rangle$ 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 $\langle (X \rightarrow Y), \leq \rangle$ 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， $f : X \rightarrow Y$ 是函数。 f 是单调的当且仅当 $\forall x_1, x_2 \in X. (x_1 \leq_1 x_2 \rightarrow f(x_1) \leq_2 f(x_2))$ 。

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 $\langle (X \rightarrow Y), \leq_2 \rangle$ 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， $f : X \rightarrow Y$ 是函数。 f 是连续的(Scott-Continuous)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 $[X \rightarrow Y]$ 。

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 $[X \rightarrow Y]$ 构成 $\langle (X \rightarrow Y), \leq \rangle$ 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， $f : X \rightarrow Y$ 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 $S \subseteq X$ ， $f(\sqcup S) \leq \sqcup f(S)$ 。

设 $\langle X, \leq \rangle$ 是偏序, $f: X \rightarrow X$ 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, $f: X \rightarrow X$ 是连续函数。则 f 有最小不动点, 且可表示如下(Kleene不动点定理)。

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

设 $\langle X, \leq \rangle$ 是完全偏序, $f: X \rightarrow X$ 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, $\varphi: X \rightarrow \{0, 1\}$ 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, $\varphi: X \rightarrow \{0, 1\}$ 是一个相容谓词。以下为不动点归纳推理(Scott Induction)。

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

一个偏序 $\langle X, \leq \rangle$, 如果 X 中的任意两个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 X 中的任意子集都有最小上界, 则称 $\langle X, \leq \rangle$ 为完全格。对称地, 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, $f: X \rightarrow X$ 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, 且可表示如下(Knaster-Tarski不动点定理)。

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

§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 。关系 \rightarrow 的传递闭包和传递自反闭包分别记做 \rightarrow^+ 和 \rightarrow^* 。

设有两个图 $D = \langle V, E \rangle$ 和 $D' = \langle V', E' \rangle$ 。若 $V \subseteq V'$ 且 $E \subseteq E'$, 则称 D 为 D' 的子图, 并表示为 $D \subseteq D'$ 。若 $V \subseteq V'$ 且 $E = \{(u, v) \in E' \mid u, v \in V\}$, 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, \dots, e_n) 称为一条通路, 其中 $e_i = \langle v_i, v_{i+1} \rangle \in E$ ($i = 1, \dots, n$)。通路 (e_1, \dots, e_n) 的长度为出现在通路中的边的次数, 记作 $|(e_1, \dots, e_n)|$ 。通路 (e_1, \dots, e_n) 通常也用顶点序列 (v_1, \dots, 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。如果回路的长度为2, 则称该回路为回路起点的顶点的自环。如果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。如果一条通路的终点与通路上的某个其它位置的顶点相同，则称该通路为套索。

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 $u \xrightarrow{+} v$ 。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。称强连通图为非平凡的，当且仅当该图至少有两个顶点或有一个有自环的顶点。

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 一步可达的顶点的集合为 $\{b \mid a \rightarrow b, a \in A\}$ ，记为 $E(A)$ 。从集合 A 可达的顶点的集合（包括 A ）为 $\bigcup_{i \geq 0} E^i(A) = \{b \mid a \xrightarrow{*} b, a \in A\}$ ，记为 $E^*(A)$ 。一步可达 A 的顶点的集合为 $\bigcup_{i \geq 0} (E^{-1})^i(A) = \{b \mid b \rightarrow a, a \in A\}$ ，记为 $E^{-1}(A)$ 。可达 A 的顶点的集合为 $\{b \mid b \xrightarrow{*} a, a \in A\}$ ，记为 $(E^{-1})^*(A)$ 。若 D 是强连通图且 $A \subseteq V$ 非空，则 $E^*(A) = (E^{-1})^*(A) = V$ 。

§1.8 练习

1. 设 $N = \{0, 1, 2, \dots\}$ 为自然数集， $X = N \cup \{\omega, \omega'\}$ 为自然数集加两个无穷大量。 X 中元素的大小满足通常对整数大小和对无穷大量的理解且定义两个无穷大量满足 $\omega < \omega'$ 。设 $f_0, f_1 : X \rightarrow X$ 定义如下： $f_0(\omega) = \omega$ ， $f_0(\omega') = \omega'$ 且若 $a \in N$ 则 $f_0(a) = a + 1$ ； $f_1(\omega) = \omega'$ ， $f_1(\omega') = \omega'$ 且若 $a \in N$ 则 $f_1(a) = a + 1$ 。 f_0 和 f_1 是单调的。证明： f_0 是连续的， f_1 不是连续的。
2. 设 $\langle X, \leq_1 \rangle$ 和 $\langle Y, \leq_2 \rangle$ 是完全偏序。 $f : X \rightarrow Y$ 是函数。证明：如果 X 只包含有穷链，则 f 是连续的当且仅当 f 是单调的。
3. 设 $D = \langle V, E \rangle$ 为有向图。设 $A \subseteq V$ 。将 $A \cup E(Z)$ 和 $A \cup E^{-1}(Z)$ 看成是自变量为 Z 的 $(2^V \rightarrow 2^V)$ 中的函数，证明这两个函数在偏序 $(2^V, \subseteq)$ 中是单调递增的， $E^*(A) = \mu Z.(A \cup E(Z))$ 且 $(E^{-1})^*(A) = \mu Z.(A \cup E^{-1}(Z))$ 。
4. 设 $D = \langle V, E \rangle$ 为有向图。证明： $E^{-1}(E(V)) = V$ 当且仅当 E 是完全关系，即任意顶点有后继。

§2 基于状态变迁的迁移系统

本章介绍以抽象状态的迁移为特点的模型。模型的基本要素为抽象状态和状态迁移关系以及初始状态。本章的算法仅限于有穷状态系统，而推理方法则不限于有穷状态系统。

基本符号： 设 S 是个集合。

S 的幂集记作 2^S 。

无穷序列 $s_0s_1s_2\dots$ 记作 $[s_i]_{i \geq 0}$ 。

有穷序列 $s_0s_1s_2\dots s_n$ 记作 $[s_i]_{i=0}^n$ 。

S^ω 表示 S 上的无穷序列的集合，即 $S^\omega = \{[s_i]_{i \geq 0} \mid \forall i \geq 0, s_i \in S\}$ 。

依上下文关系， S^n 可表示 S 上的 n 元组的集合 $\{(s_1, \dots, s_n) \mid s_1, \dots, s_n \in S\}$ 或长度为 n 的有穷序列的集合 $\{[s_i]_{i=0}^{n-1} \mid s_0, \dots, s_{n-1} \in S\}$ 。

若 $\rightarrow \subseteq S \times S$ 为 S 上的二元关系，则 \rightarrow^* 表示 \rightarrow 的传递自反闭包。

集合的运算包括交、并、差，记作 \cap, \cup, \setminus ，为方便起见，有时写作 $*, +, -$ 。

设 S 为全集。集合 A 的补为全集与 A 的差 $S \setminus A$ ，亦记作 $\neg A$ 。

若 $Y \subseteq S \times S$ 且 $X \subseteq S$ ，用 $Y|X$ 表示 $Y \cap (X \times X)$ 。

若 $G = (V, E)$ 是有向图，依上下文关系，为方便起见， V 的子集 B 亦可表示顶点集合为 B 的导出子图 $(B, E|B)$ 。

用 e_1, \dots, e_k 分别替换 φ 中的 x_1, \dots, x_k 记作 $\varphi(e_1/x_1, \dots, e_k/x_k)$ ，亦写作 $\varphi_{x_1, \dots, x_k}^{e_1, \dots, e_k}$ 。

§2.1 Kripke模型

定义 2.1 一个Kripke模型是一个三元组 $\langle S, R, I \rangle$ 其中 S 为状态集合， $R \subseteq S \times S$ 为 S 上的完全迁移关系， $I \subseteq S$ 为初始状态集。

$R \subseteq S \times S$ 是完全的，即 $\forall s. \exists s'. (s, s') \in R$ 。

我们也称Kripke模型为系统。

用 $s \rightarrow s'$ 表示存在从 s 到 s' 的迁移，即 $(s, s') \in R$ 。

假定 $K = \langle S, R, I \rangle$ 为给定模型。

后继状态： 若 $s \rightarrow s'$ ，则称 s' 为 s 的后继状态。状态 s 的后继状态的集合为 $\{s' \mid s \rightarrow s'\}$ ，记作 $R(s)$ 。状态集合 A 的后继状态集合为 $\bigcup_{s \in A} R(s)$ ，记作 $R(A)$ 。

前驱状态： 若 $s \rightarrow s'$ ，则称 s 为 s' 的前驱状态。状态 s 的前驱状态的集合为 $\{s' \mid s' \rightarrow s\}$ ，记作 $R^{-1}(s)$ 。状态集合 A 的前驱状态集合为 $\bigcup_{s \in A} R^{-1}(s)$ ，记作 $R^{-1}(A)$ 。

可达状态： 若 $s \xrightarrow{*} s'$ ，则称 s' 为 s 的可达状态。状态 s 的可达状态的集合为 $\{s' \mid s \xrightarrow{*} s'\}$ ，记作 $R^*(s)$ 。状态集合 A 的可达状态集合为 $\bigcup_{s \in A} R^*(s)$ ，记作 $R^*(A)$ 。 K 的可达状态集合为 $R^*(I)$ ，亦记作 $Rh(K)$ 。

可达关系： 若 B 中有 s 的可达状态，则称 B 由 s 可达。若 B 中有 A 的可达状态，则称 B 由 A 可达，记作 $A \xrightarrow{*} B$ 。

路径： K 的有穷路径是 S 上的满足对任意 $0 \leq i \leq n-1$ 都有 $s_i \rightarrow s_{i+1}$ 的有穷序列 $[s_i]_{i=0}^n$ 。 K 的无穷路径是 S 上的满足对任意 $i \geq 0$ 都有 $s_i \rightarrow s_{i+1}$ 的无穷序列 $[s_i]_{i \geq 0}$ 。

计算与行为： K 的计算序列(简称为计算)是 K 上的满足 $s_0 \in I$ 的无穷路径 $[s_i]_{i \geq 0}$ 。 K 的计算的集合称为 K 的行为, 记作 $[[K]]$ 。

状态的满足关系： 称状态 s 满足 A , 当且仅当 $s \in A$ 。因而集合 A 中的状态都满足 A , 且满足 A 的状态都在 A 中。称集合 B 满足 A , 当且仅当 B 中的所有状态都满足 A , 即 $B \subseteq A$ 。为方便叙述, 满足 A 的状态称为 A 状态。若一条路径(或一个计算)上的所有状态都满足 A , 则称该路径(计算)为 A 路径(计算)。若一条路径(或一个计算)上有满足 A 的状态, 则称该路径(计算)能到达 A 状态。

§2.1.1 可达性质分析

定义 2.2 (可达性质) 状态集合 $A \subseteq S$ 称为 K 的可达性质, 当且仅当 $[[K]]$ 中有能到达 A 状态的计算。

命题 2.1 状态集合 $A \subseteq S$ 是 K 的可达性质当且仅当 $Rh(K) \cap A \neq \emptyset$ 。

给定 Kripke 模型 $K = \langle S, R, I \rangle$ 和集合 $A \subseteq S$ 。 A 是否是 K 的可达性质可通过对模型可达状态的检查来验证。

可达性质分析1： 首先介绍基于状态遍历的可达性质分析算法。为方便算法设计, 我们定义集合类型结构的操作。我们记空集为 $\{\}$ 。设 w 为集合。用 $w.put(s)$, $w.get()$, $w.remove(s)$ 分别代表在集合 w 中添加元素 s , 从集合中读取元素, 从集合中移除元素 s 。给定 Kripke 模型 $K = \langle S, R, I \rangle$ 和集合 $A \subseteq S$ 。设每个状态 s 有 $s.visited$ 这个属性, 其初始值为 $false$ 。算法1是基于状态遍历的分析 A 是否为可达性质的算法。

Algorithm 1 ReachabilityAnalysis

Input: K, A ;

Output: $true/false$;

```

1:  $w := I$ ;
2: while ( $w \neq \emptyset$ ) do
3:    $s := w.get()$ ; if ( $s \in A$ ) return true;
4:    $s.visited := true$ ;
5:   for each ( $s' \in R(s)$ ), if ( $s'.visited = false$ )  $w.put(s')$ ;
6:    $w.remove(s)$ ;
7: end while
8: return false;
```

命题 2.2 状态集合 $A \subseteq S$ 是 K 的可达性质当且仅当 $ReachabilityAnalysis(K, A)$ 为 $true$ 。

可达性质分析2： 以下是基于不动点计算的可达性质分析。首先介绍有穷集合完全格上的单调函数的不动点算法如下。

最小和最大不动点的计算： 设 X 为有穷集合， (X, \sqsubseteq) 为完全格， f 为 X 上的单调函数。记最小元为 \perp ，最大元为 \top 。算法2计算 f 的以元素 A 为初始值的不动点。若 A 为 \perp 时，计算得到的为 f 的最小不动点，若 A 为 \top 时，计算得到的为 f 的最大不动点，即我们有 f 的最小不动点 $\mu f = \text{FNfp}(f, \perp)$ 和 f 的最大不动点 $\nu f = \text{FNfp}(f, \top)$ 。

Algorithm 2 FNfp

Input: 函数 f 和元素 A ;

Output: 函数 f 以 A 为初始值的不动点;

```

1:  $X := A; Y := f(A)$ ;
2: while  $X \neq Y$  do
3:    $X := Y; Y := f(X)$ ;
4: end while
5: return  $X$ ;

```

基于最小不动点计算的可达性质分析的算法： 给定Kripke模型 $K = \langle S, R, I \rangle$ 和集合 $A \subseteq S$ 。设 S 为有穷状态集合。则 $(2^S, \subseteq)$ 为完全偏序且空集为偏序的最小元。定义 $f_K : 2^S \rightarrow 2^S$ 如下： $f_K(A) = I \cup R(A)$ 。则 f_K 为 $(2^S, \subseteq)$ 上的单调函数且以下命题成立。

命题 2.3 $Rh(K) = \mu f_K$.

由命题2.3和命题2.1，知算法3是基于不动点计算的分析 A 是否为可达性质的算法。

Algorithm 3 ReachabilityAnalysisFP

Input: K, A ;

Output: *true/false*;

```

1:  $w := \text{FNfp}(f_K, \emptyset)$ ;
2: return  $w \cap A \neq \emptyset$ ;

```

命题 2.4 状态集合 $A \subseteq S$ 是 K 的可达性质当且仅当 $\text{ReachabilityAnalysisFP}(K, A)$ 为*true*。

§2.1.2 安全性质分析

定义 2.3 (安全性质) 状态集合 $A \subseteq S$ 称为 K 的安全性质，当且仅当 K 的所有计算都是 A 计算。

命题 2.5 状态集合 $A \subseteq S$ 是 K 的安全性质当且仅当 $Rh(K) \subseteq A$ 。

推论 2.1 状态集合 $A \subseteq S$ 是 K 的安全性质，当且仅当状态集合 $\neg A$ 不是 K 的可达性质。

安全性质分析算法： 根据推论2.1知安全性质与可达性质是对偶性质。我们可以对可达性质分析算法做简单修改以分析安全性质。

安全性质分析算法1： 算法4是基于状态遍历的分析 A 是否为安全性质的算法。

命题 2.6 状态集合 $A \subseteq S$ 是 K 的安全性质当且仅当 $\text{SafetyAnalysis}(K, A)$ 为*true*。

Algorithm 4 SafetyAnalysis

Input: K, A ;**Output:** $true/false$;

```
1:  $w := I$ ;  
2: while ( $w \neq \emptyset$ ) do  
3:    $s := w.get()$ ; if ( $s \notin A$ ) return  $false$ ;  
4:    $s.visited := true$ ;  
5:   for each ( $s' \in R(s)$ ), if ( $s'.visited = false$ )  $w.put(s')$ ;  
6:    $w.remove(s)$ ;  
7: end while  
8: return  $true$ ;
```

Algorithm 5 SafetyAnalysisFP

Input: K, A ;**Output:** $true/false$;

```
1:  $w := FNfp(f_K, \emptyset)$ ;  
2: return  $w \cap \neg A = \emptyset$ ;
```

安全性质分析算法2: 算法5是基于最小不动点计算的分析A是否为安全性质的算法。

命题 2.7 状态集合 $A \subseteq S$ 是 K 的安全性质当且仅当 $SafetyAnalysisFP(K, A)$ 为 $true$ 。

由于安全性质与可达性质是对偶性质，我们有以下结论。

推论 2.2 $SafetyAnalysis(K, A)$ 为 $true$ 当且仅当 $ReachabilityAnalysis(K, \neg A)$ 为 $false$ 。

安全性质的推理: 若 $R(A) \subseteq A$ ，则称 A 为迁移不变。关于安全性质，我们有如下推理规则。设 A', A 为状态集合。

$$\frac{\begin{array}{l} I \text{ 满足 } A' \\ A' \text{ 迁移不变} \\ A' \text{ 满足 } A \end{array}}{A \text{ 是安全性质}}$$

可靠性与完备性: 以上推理规则是可靠的且完备的。假定前提成立，由前两条我们知道 $R^*(I) \subseteq A'$ ，又由第三条我们知道 $R^*(I) \subseteq A$ ，所以 A 是安全性质，因而推理规则是可靠的。假定 A 是安全的，取 $A' = R^*(I)$ ，我们就可保证前提条件的成立，由此可证 A 是安全性质，因而推理规则是完备的。

§2.1.3 可避免性质分析

定义 2.4 (可避免性质) 状态集合 $A \subseteq S$ 称为 K 的可避免性质，当且仅当 $[[K]]$ 中有 $\neg A$ 计算。

命题 2.8 状态集合 $A \subseteq S$ 是 K 的可避免性质当且仅当 $\exists B \subseteq S. ((B \cap I \neq \emptyset) \wedge (\forall s \in B. (\exists s' \in B. (s \rightarrow s'))) \wedge (B \cap A = \emptyset))$ 。

可避免性质分析1: 给定Kripke模型 $K = \langle S, R, I \rangle$ 和集合 $A \subseteq S$ 。A是否是K的可避免性质可通过对模型的强连通图的分析来验证,即考察模型是否可由 $\neg A$ 有穷路径到达由 $\neg A$ 状态组成的非平凡强连通图。

强连通分图的计算: 为方便算法设计,我们定义栈类型结构的操作。对于栈类型的变量,我们记空栈为 $[]$ 。设 w 为栈。用 $w.push(s)$, $w.top()$, $w.pop()$ 分别代表在栈 w 的栈顶添加元素 s ,从栈中读取栈顶元素,从栈中移除栈顶元素。给定一个有向图 $G = (V, E)$ 。设每个顶点 v 有 $v.index$ 和 $v.lowlink$ 两个属性,其初始值为 $undefined$ 。算法6是计算强连通分图的Tarjan算法。算法返回值 $FNscc(G)$ 是G的强连通分图顶点集合的集合。

Algorithm 6 $FNscc$

Input: $G = (V, E)$;

Output: G 的强连通分图顶点集合的集合;

```

1:  $index := 0$ ;  $S := []$ ;  $scclist := \{\}$ ;
2: for each  $v \in V$ , if ( $v.index = undefined$ ) strongconnect( $v$ );
3: return  $scclist$ ;
4: FUNCTION strongconnect( $v$ )
5:  $v.index := v.lowlink := index$ ;  $index := index + 1$ ;  $S.push(v)$ ;
6: for each  $w \in E(v)$  do
7:   if ( $w.index = undefined$ ) then
8:     strongconnect( $w$ );  $v.lowlink := \min(v.lowlink, w.lowlink)$ ;
9:   else if ( $w \in S$ )
10:     $v.lowlink := \min(v.lowlink, w.index)$ ;
11:   end if
12: end for
13: if ( $v.lowlink = v.index$ ) then
14:    $currentSCC := \{\}$ ;
15:   while true do
16:     $w := S.top()$ ;  $S.pop()$ ;  $currentSCC.put(w)$ ; if ( $w = v$ ) break;
17:   end while
18:    $scclist.put(currentSCC)$ ;
19: end if

```

可避免性质分析算法1: 假定 $nontrivial(G, e)$ 为检查 G 的强连通分量 e 是否非平凡的函数。给定 $K = \langle S, R, I \rangle$ 和集合 $A \subseteq S$ 。定义 $K|X = \langle S \cap X, R|X, I \cap X \rangle$ 。A是可避免性质当且仅当在 $K| \neg A$ 中存在一条由初始状态出发的无穷路径。算法7是基于强连通分量计算的分析A是否为可避免性质的算法。

命题 2.9 状态集合 $A \subseteq S$ 是K的可避免性质当且仅当 $AvoidabilityAnalysis(K, A)$ 为true。

可避免性质分析2: 以下是基于不动点计算的可避免性质分析。给定有向图 $G = (S, R)$ 且 $A \subseteq S$ 。设 S 的有穷集合。则 $(2^S, \subseteq)$ 为完全偏序且 S 为偏序的最大元。定义 $f_G : 2^S \rightarrow 2^S$ 如下: $f_G(A) =$

Algorithm 7 AvoidabilityAnalysis

Input: K, A ;**Output:** $true/false$;

- 1: $K' := (S', R', I') := K|\neg A; G := (S', R')$;
 - 2: $scclist := FNsccl(G)$;
 - 3: $w := \emptyset$; for (each e in $scclist$) if ($nontrivial(G, e)$) $w := (w \cup e)$;
 - 4: return $ReachabilityAnalysis(K', w)$;
-

$R^{-1}(A)$ 。则 f_G 为 $(2^S, \subseteq)$ 上的单调函数且以下命题成立。

命题 2.10 v_f 为 G 中所有无穷路径起点状态组成的集合。

可避免性质分析算法2: 由命题2.10知算法8是基于最大不动点计算的分析 A 是否为可避免性质的算法。

Algorithm 8 AvoidabilityAnalysisFP

Input: K, A ;**Output:** $true/false$;

- 1: $G := (\neg A, R|\neg A)$;
 - 2: $w := FNfp(f_G, \neg A)$;
 - 3: return $w \cap I \neq \emptyset$;
-

命题 2.11 状态集合 $A \subseteq S$ 是 K 的可避免性质当且仅当 $AvoidabilityAnalysisFP(K, A)$ 为 $true$ 。

§2.1.4 必达性质分析

定义 2.5 (必达性质) 状态集合 $A \subseteq S$ 称为 K 的必达性质，当且仅当 K 的所有计算都能到达 A 状态。

命题 2.12 状态集合 $A \subseteq S$ 是 K 的必达性质当且仅当 $\forall B \subseteq S. ((B \cap I \neq \emptyset) \wedge (\forall s \in B. (\exists s' \in B. (s \rightarrow s'))) \rightarrow (B \cap A \neq \emptyset))$ 。

推论 2.3 状态集合 $A \subseteq S$ 是 K 的必达性质，当且仅当状态集合 A 不是 K 的可避免性质。

必达性质分析算法: 根据推论2.3知必达性质与可避免性质是相反的关系。我们可以对可避免性质分析算法做简单修改以分析必达性质。给定 $K = \langle S, R, I \rangle$ 和集合 $A \subseteq S$ 。算法9是基于强连通分量计算的分析 A 是否为必达性质的算法。

命题 2.13 状态集合 $A \subseteq S$ 是 K 的必达性质当且仅当 $InevitabilityAnalysis(K, A)$ 为 $true$ 。

推论 2.4 $InevitabilityAnalysis(K, A)$ 为 $true$ 当且仅当 $AvoidabilityAnalysis(K, A)$ 为 $false$ 。

Algorithm 9 InevitabilityAnalysis

Input: K, A ;**Output:** $true/false$;

- 1: $K' := (S', R', I') := K|\neg A; G := (S', R')$;
 - 2: $scclist := FNsccl(G)$;
 - 3: $w := \emptyset$; for (each e in $scclist$) if ($nontrivial(G, e)$) $w := (w \cup e)$;
 - 4: return not $ReachabilityAnalysis(K', w)$;
-

必达性质的推理: 我们有如下推理规则。设 X_0, X_1, \dots, X_n, A 为状态集合。

$$\begin{array}{l} I \text{ 满足 } X_0 \\ \forall i \in \{0, \dots, n-1\}, R(X_i \setminus A) \text{ 满足 } X_{i+1} \\ X_n \text{ 满足 } A \\ \hline A \text{ 是必达性质} \end{array}$$

可靠性与完备性: 以上推理规则是可靠的且对于有穷状态系统是完备的。假定前提成立且结论不成立, 那么由结论不成立, 我们有一条由 I 状态出发的无穷 $\neg A$ 路径, 这条路径必然有状态出现在任一 X_i 中, 与 X_n 满足 A 矛盾, 所以前提成立则结论成立, 因而推理规则是可靠的。假定 A 是必达性质, 由于只考虑有穷状态系统, 我们可以假定状态个数为 n , 那么取 $X_0 = I$ 和 $X_{i+1} = R(X_i \setminus A)$, 则 X_n 为空(否则存在一条无限循环的 $\neg A$ 路径), 由此可证 A 是必达性质, 因而推理规则是完备的。

§2.2 公平Kripke模型

对于Kripke模型而言, 一个计算是以一个初始状态为起点的满足状态迁移关系的无穷状态序列。为了更精确地描述模型的计算, 我们对计算做一定限制以排除部分满足状态转换关系的无穷状态序列。需要排除的部分称为不公平的无穷状态序列。为达到这个效果, 我们在结构中加入新的成分, 设置公平计算的条件。

定义 2.6 一个公平Kripke模型是一个四元组 $\langle S, R, I, F \rangle$ 其中 $\langle S, R, I \rangle$ 是一个Kripke模型, $F \subseteq 2^S$ 为公平性约束的有穷集合。

假定 $K = \langle S, R, I, F \rangle$ 为给定模型。

可达状态: 标号Kripke模型 K 的可达状态即 $K' = \langle S, R, I \rangle$ 的可达状态。

公平路径: 公平Kripke模型 K 上的路径即Kripke模型 $\langle S, R, I \rangle$ 上的路径。定义 $inf(\pi)$ 为无限多次出现在路径 π 中的状态的集合。路径 $\pi = [s_i]_{i \geq 0}$ 是公平的, 当且仅当对所有 $f \in F$,

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

计算与行为: 公平Kripke模型 K 上的计算即Kripke模型 $\langle S, R, I \rangle$ 上的计算。一个计算是公平的, 当且仅当该计算是一条公平路径。 K 的公平计算的集合称为 K 的行为, 记作 $[[K]]$ 。

公平状态： 一个状态是公平的，当且仅当存在从该状态出发的公平路径。

公平状态集合： 记 $\{s \mid s \rightarrow s', s' \in X\}$ 为 $R^{-1}(X)$ 。

定义 $S_F = \nu Z. \bigwedge_{f \in F} \mu Y. ((f \cap R^{-1}(Z)) \cup R^{-1}(Y))$ 。

将 $((X \cap R^{-1}(Z)) \cup R^{-1}(Y))$ 看成是自变量为 Y 的从 2^S 到 2^S 函数，我们知道这个函数在完备格 $(2^S, \subseteq)$ 上是单调递增的，有最小不动点，因而 $\bigwedge_{f \in F} \mu Y. ((f \cap R^{-1}(Z)) \cup R^{-1}(Y))$ 是良定义。将 $\bigwedge_{f \in F} \mu Y. ((f \cap R^{-1}(Z)) \cup R^{-1}(Y))$ 看成是自变量为 Z 的函数，我们知道这个函数是单调递增的，有最大不动点，因而 S_F 是良定义。

命题 2.14 状态 $s \in S$ 是 K 的公平状态，当且仅当 $s \in S_F$ 。

可达公平状态集合： 记由 A 可达的公平状态集合为 $Rh_F(A)$ 。

命题 2.15 $Rh_F(A) = Rh(A) \cap S_F$ 。

K 的可达公平状态集合，记为 $Rh_F(A)$ ，即由 I 可达的公平状态集合 $Rh_F(I)$ 。

推论 2.5 $Rh_F(K) = Rh(I) \cap S_F$ 。

公平强连通分量： 考虑穷状态系统。给定 $K = \langle S, R, I, F \rangle$ 。有向图 (S, R) 的强连通分量也称为 K 的强连通分量。 K 的强连通分量 e 是公平的，当且仅当 e 是非平凡的且 $\forall f \in F. (e \cap f \neq \emptyset)$ ，其中 $e \cap f$ 表示 f 中的元素和强连通分量 e 中的元素的交集。算法10是检查 e 是否为公平强连通分量的算法。

Algorithm 10 Fairsc

Input: K, e ;

Output: *true/false*;

- 1: if (*nontrivial*(K, e) = *false*) return *false*;
 - 2: for (each f in F) if ($e \cap f = \emptyset$) return *false*;
 - 3: return *true*;
-

命题 2.16 K 的强连通分量 e 是公平的，当且仅当 $Fairsc(K, e)$ 为*true*。

公平状态判定： 在公平强连通分量检查算法的基础上，可以设计检查一个状态是否为公平状态和一个状态集中是否有公平状态的算法。算法11是检查 A 中是否有公平状态的算法。

命题 2.17 状态集 $A \subseteq S$ 中有公平状态，当且仅当 $ExistFairState(K, A)$ 为*true*。

推论 2.6 状态 $s \in S$ 是 K 的公平状态，当且仅当 $ExistFairState(K, \{s\})$ 为*true*。

Algorithm 11 ExistFairState

Input: K, A ;**Output:** $true/false$;

- 1: $G := (S, R)$; $scclist := FNscg(G)$;
 - 2: $w := \emptyset$; for (each e in $scclist$) if ($fairscg(K, e)$) $w := w \cup e$;
 - 3: $K' := (S, R, A)$;
 - 4: return $ReachabilityAnalysis(K', w)$;
-

§2.2.1 公平可达性质分析

定义 2.7 (公平可达性质) 状态集合 $A \subseteq S$ 称为 K 的公平可达性质, 当且仅当 $[[K]]$ 中有能到达 A 状态的公平计算。

命题 2.18 状态集合 $A \subseteq S$ 是 K 的公平可达性质当且仅当 $Rh_F(K) \cap A \neq \emptyset$ 。

推论 2.7 状态集合 $A \subseteq S$ 是 K 的公平可达性质当且仅当 $Rh(K) \cap A$ 中存在公平状态。

可达性分析1: 给定公平Kripke模型 $K = \langle S, R, I, F \rangle$ 和集合 $A \subseteq S$ 。 A 是否是 K 的公平可达性质的分析可通过首先看哪些 A 状态是公平状态, 再结合可达性分析算法看这些公平状态中是否有由模型初始状态可达的。算法12是根据原来的可达性分析算法修改的公平可达性分析算法。

Algorithm 12 FairReachabilityAnalysis

Input: K, A ;**Output:** $true/false$;

- 1: $w := I$;
 - 2: **while** ($w \neq \emptyset$) **do**
 - 3: $s := w.get()$; if ($s \in A$) and ($ExistFairState(K, \{s\}) = true$) return $true$;
 - 4: $s.visited := true$;
 - 5: for each ($s' \in R(s)$), if ($s'.visited = false$) $w.put(s')$;
 - 6: $w.remove(s)$;
 - 7: **end while**
 - 8: return $false$;
-

命题 2.19 状态集合 $A \subseteq S$ 是 K 的公平可达性质当且仅当 $FairReachabilityAnalysisII(K, A)$ 为 $true$ 。

可达性分析2: 由于 $FairReachabilityAnalysis()$ 算法每次碰到一个 A 状态都要调用一次 $ExistFairState()$, 我们可以考虑收集这些可达的 A 状态, 然后一次性地看其中是否有公平状态。算法13是根据这个结论和原来的可达性分析算法修改的公平可达性分析算法。

命题 2.20 状态集合 $A \subseteq S$ 是 K 的公平可达性质当且仅当 $FairReachabilityAnalysisII(K, A)$ 为 $true$ 。

Algorithm 13 FairReachabilityAnalysisII

Input: K, A ;**Output:** $true/false$;

```
1:  $w := I; u := \emptyset$ ;  
2: while ( $w \neq \emptyset$ ) do  
3:    $s := w.get()$ ; if ( $s \in A$ )  $u.put(s)$ ;  
4:    $s.visited := true$ ;  
5:   for each ( $s' \in R(s)$ ), if ( $s'.visited = false$ )  $w.put(s')$ ;  
6:    $w.remove(s)$ ;  
7: end while  
8: return  $ExistFairState(K, u)$ ;
```

§2.2.2 公平安全性质分析

定义 2.8 (公平安全性质) 状态集合 $A \subseteq S$ 称为 K 的公平安全性质, 当且仅当 K 的所有公平计算都是 A 计算。

命题 2.21 状态集合 $A \subseteq S$ 是 K 的公平安全性质, 当且仅当以下之一成立。

- 状态集合 $\neg A$ 不是 K 的公平可达性质;
- $Rh_F(K) \subseteq A$;
- $FairReachabilityAnalysis(K, \neg A)$ 为 $false$ 。

公平安全性质的推理: 给定 $K = \langle S, R, I, F \rangle$ 且 $F = \{f_1, \dots, f_k\}$ 。设 X_0, X_1, \dots, X_k, A 为状态集。我们有以下规则。

$$\frac{\begin{array}{l} A \cup X_0 \text{ 是 } \langle S, R, I \rangle \text{ 的安全性质} \\ \bigcup_{i=1}^k X_i \text{ 是 } \langle S, R, X_0 \rangle \text{ 的必达性质} \\ \forall i \in \{1, \dots, k\}, \neg f_i \text{ 是 } \langle S, R, X_i \rangle \text{ 的安全性质} \end{array}}{A \text{ 是 } K \text{ 的公平安全性质}}$$

可靠性: 假定前提成立。由前提3知, 对 $i = 1, \dots, k$, X_i 状态以及 X_i 可达的状态都不是公平状态。由于 X_0 不能避开 X_1, \dots, X_k , 因而 X_0 状态不是公平状态。由于 $A \cup X_0$ 是 $\langle S, R, I \rangle$ 的安全性质, 因而所有 K 的公平计算都是 A 计算, 即 A 是 K 的公平安全性质。由于在以 X_0 状态不是公平状态为前提条件的情况下, 并不一定保证存在 X_1, \dots, X_k 使得前提2和前提3成立, 因而推理规则不具有完备性。

§2.2.3 公平可避免性质分析

定义 2.9 (公平可免性质) 状态集合 $A \subseteq S$ 称为 K 的公平可免性质, 当且仅当 $[[K]]$ 中有公平 $\neg A$ 计算。

考虑有穷状态系统。用 $scc_F(B)$ 表示 $B \subseteq S$ 为公平强连通图，即 B 是非平凡强连通的且 $\forall f \in F.(B \cap f \neq \emptyset)$ 。

命题 2.22 状态集合 $A \subseteq S$ 是 K 的公平可免性质当且仅当 $\exists B \subseteq S.((B \cap I \neq \emptyset) \wedge \exists B' \subseteq B.(scc_F(B') \wedge \forall B'' \subseteq (B \setminus B').(\exists s \in B''.(B'' \neq \emptyset \rightarrow \exists s' \in (B \setminus B'').(s \rightarrow s')))) \wedge (B \cap A = \emptyset))$ 。

公平可避免性分析1: 给定公平Kripke模型 $K = \langle S, R, I, F \rangle$ 和集合 $A \subseteq S$ 。 A 是否是 K 的公平可免性分析类似于原先的可免性分析，只是将所用到的非平凡强连通分量加强为公平强连通分量即可。定义 $\{f_1, \dots, f_n\} \parallel Y = \{f_1 \cap Y, \dots, f_n \cap Y\}$ 和 $\langle S, R, I, F \rangle \parallel X = \langle S \cap X, R \parallel X, I \cap X, F \parallel X \rangle$ 。算法14是分析 A 是否为公平可避免性质的算法。

Algorithm 14 FairAvoidabilityAnalysis

Input: K, A ;

Output: *true/false*;

- 1: $K' := (S', R', I', F') := K \upharpoonright \neg A$; $G := (S', R')$; $K'' := (S', R', I')$;
 - 2: $scclist := FNsccl(G)$;
 - 3: $w := \emptyset$; for (each e in $scclist$) if ($Fairsccl(K', e)$) $w := (w \cup e)$;
 - 4: return $ReachabilityAnalysis(K'', w)$;
-

命题 2.23 状态集合 $A \subseteq S$ 是 K 的公平可免性质当且仅当 $FairAvoidabilityAnalysis(K, A)$ 为 *true*。

公平可避免性分析2: 由于 A 是公平可免性质当且仅当 $K \upharpoonright \neg A$ 有公平可达状态当且仅当 $K \upharpoonright \neg A$ 的初始状态中有公平可达状态，使用 $ExistFairState()$ ，则公平可免性分析可以更加简单。

命题 2.24 状态集合 $A \subseteq S$ 是 K 的公平可免性质当且仅当 $ExistFairState(K \upharpoonright \neg A, I \setminus A)$ 为 *true*。

§2.2.4 公平必达性质分析

定义 2.10 (公平必达性质) 状态集合 $A \subseteq S$ 称为 K 的公平必达性质，当且仅当 K 的所有公平计算都能到达 A 状态。

命题 2.25 状态集合 $A \subseteq S$ 是 K 的公平必达性质，当且仅当以下之一成立。

- 状态集合 A 不是 K 的公平可避免性质;
- 对所有含有 K 的初始状态的 S 的子集 B 以下成立:

$$(\exists B' \subseteq B.(scc_F(B') \wedge \forall B'' \subseteq (B \setminus B'').(B'' \neq \emptyset \rightarrow \exists s \in B''.(\exists s' \in (B \setminus B'').(s \rightarrow s')))) \rightarrow (B \cap A \neq \emptyset));$$

- $FairAvoidability(K, A)$ 为 *false*;
- $ExistFairState(K \upharpoonright \neg A, I \setminus A)$ 为 *false*。

公平必达性质的推理： 给定 $K = \langle S, R, I, F \rangle$ 且 $F = \{f_1, \dots, f_k\}$ 。设 X_0, X_1, \dots, X_k, A 为状态集。我们有以下规则。

$$\begin{array}{l} A \cup X_0 \text{ 是 } \langle S, R, I \rangle \text{ 的必达性质} \\ \bigcup_{i=1}^k X_i \text{ 是 } \langle S, R, X_0 \rangle \text{ 的必达性质} \\ \forall i \in \{1, \dots, k\}, \neg f_i \text{ 是 } \langle S, R, X_i \rangle \text{ 的安全性质} \\ \hline A \text{ 是 } K \text{ 的公平必达性质} \end{array}$$

可靠性： 假定前提成立，那么 X_0 状态不是公平状态。由于 $A \cup X_0$ 是 $\langle S, R, I \rangle$ 的必达性质，因而所有 K 的公平计算都必然可达 A 状态，即 A 是 K 的公平必达性质。

§2.2.5 模型非空问题

由于对计算增加了限制，有可能使得所有计算都成了不公平的计算，这样，模型的行为就是一个空集。我们可以检查模型的行为是否为空。这个问题也称为模型非空问题。模型 K 非空记作 $K \neq \emptyset$ ，当且仅当 $[[K]] \neq \emptyset$ 。关于模型非空问题，我们有以下结论。

命题 2.26 给定 $K = \langle S, R, I, F \rangle$ 。 $K \neq \emptyset$ 当且仅当存在由 I 可达的公平强连通分量。

推论 2.8 给定 $K = \langle S, R, I, F \rangle$ 。 $K \neq \emptyset$ 当且仅当 I 中存在公平状态。

模型空性检查算法1： 检查模型是否为空的算法称为空性检查算法。给定 $K = \langle S, R, I, F \rangle$ 。算法15是基于以上命题的空性检查算法。

Algorithm 15 EmpChecking

Input: K ;

Output: *true/false*;

- 1: $G := (S, R)$; $scclist := FNsc(G)$;
 - 2: $w := \emptyset$; for (each e in $scclist$) if ($fairsc(K, e)$) $w := w \cup e$;
 - 3: $K' := (S, R, I)$;
 - 4: return not $ReachabilityAnalysis(K', w)$;
-

命题 2.27 $K = \emptyset$ 当且仅当 $EmpChecking(K)$ 为*true*。

又由于 $K \neq \emptyset$ 当且仅当 I 中存在公平状态。我们由以下结论。

命题 2.28 $K = \emptyset$ 当且仅当 $ExistFairState(K, I)$ 为*false*。

模型空性检查算法2： 若 $|F| = 1$ ，即 F 中只有一个元素，则我们可以用一个双深度优先算法计算模型是否为空问题。给定 $K = \langle S, R, I, \{f\} \rangle$ 。算法16是这样的一个算法。

命题 2.29 给定 $K = \langle S, R, I, F \rangle$ 且 $|F| = 1$ 。 $K = \emptyset$ 当且仅当 $EmpCheckingDDFS(K)$ 为*true*。

§2.2.6 非空问题的应用

非空问题是一个基本问题，可应用于解决其它的问题。

Algorithm 16 EmpCheckingDDFS

Input: K ;**Output:** $true/false$;

```
1:  $w := []$ ;  $a := b := \{\}$ ;
2: for each  $s \in I$  do
3:   if ( $s \notin a$ ) then
4:      $a.put(s)$ ;
5:      $w.push(s)$ ; if (empcheckingDFS1( $s$ )= $false$ ) return  $false$ ;  $w.pop()$ ;
6:   end if
7: end for
8: return  $true$ ;
9: FUNCTION empcheckingDFS1( $v$ )
10: for each  $s \in R(v)$  do
11:   if ( $s \notin a$ ) then
12:      $a.put(s)$ ;
13:      $w.push(s)$ ; if (empcheckingDFS1( $s$ )= $false$ ) return  $false$ ;  $w.pop()$ ;
14:   end if
15: end for
16: if ( $v$  in  $f$ ) {  $b.put(s)$ ; if (empcheckingDFS2( $s$ )= $false$ ) return  $false$ ; }
17: return  $true$ ;
18: FUNCTION empcheckingDFS2( $v$ )
19: for each  $s \in R(v)$  do
20:   if ( $s$  in  $w$ ) return  $false$ ;
21:   if not ( $s$  in  $b$ ) {  $b.put(s)$ ; if (empcheckingDFS2( $s$ )= $false$ ) return  $false$ ; };
22: end for
23: return  $true$ ;
```

可达性问题 可达性问题可以转换到非空问题。给定Kripke模型 $K = \langle S, R, I \rangle$ 和 $A \subseteq S$ 。定义 $K' = \langle S', R', I', F \rangle$ 其中 S', R', I', F 如下。

$$\begin{aligned} S' &= S \cup \{t\} \\ R' &= R \cup \{(s, t) \mid s \in A\} \cup \{(t, t)\} \\ I' &= I \\ F &= \{\{t\}\} \end{aligned}$$

命题 2.30 状态集合 $A \subseteq S$ 是 K 的可达性质, 当且仅当 K' 非空, 当且仅当 $EmpCheckingDDFS(K')$ 为 $false$ 。

可避免性问题 可避免性问题可以转换到非空问题。给定Kripke模型 $K = \langle S, R, I \rangle$ 和 $A \subseteq S$ 。定义 $K' = \langle S', R', I', F \rangle$ 其中 S', R', I', F 如下。

$$\begin{aligned} S' &= S \cup \{t\} \\ R' &= \{(s, s') \mid (s, s') \in R, s \notin A\} \cup \{(s, t) \mid s \in A\} \cup \{(t, t)\} \\ I' &= I \\ F &= \{S\} \end{aligned}$$

命题 2.31 状态集合 $A \subseteq S$ 是 K 的可避免性质, 当且仅当 K' 非空, 当且仅当 $EmpCheckingDDFS(K')$ 为 $false$ 。

§2.2.7 强公平与弱公平条件

我们对公平 Kripke 模型, 可以增强其公平条件。

强公平 Kripke 模型: 用 $F \subseteq 2^S \times 2^S$ 取代原来 $K = \langle S, R, I, F \rangle$ 的公平条件, 并将公平路径定义如下。一条路径 $\pi = [s_i]_{i \geq 0}$ 是公平的, 当且仅当对所有 $(f, g) \in F$,

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

按照这样的公平路径的定义, 我们有一个强公平 Kripke 模型。

强公平模型的表达能力: 强公平 Kripke 模型增强了原公平 Kripke 模型的表达能力。

- 给定公平 Kripke 模型 $K = \langle S, R, I, \{g_1, \dots, g_k\} \rangle$ 。定义 $F' = \{(S, g_1), \dots, (S, g_k)\}$ 。则强公平 Kripke 模型 $K' = \langle S, R, I, F' \rangle$ 与 K 有相同的公平计算集合。
- 强公平 Kripke 模型对原公平 Kripke 模型的增强可以用如下例子作证。设 $K_1 = \langle S, R, I, F \rangle$ 其中 S, R, I, F 定义如下。容易验证没有公平 Kripke 模型能够有与强公平 Kripke 模型 K_1 相同的公平计算集合。

$S = \{s0, s1, s2\}$ $R = \{(s0, s0), (s1, s1), (s2, s2), (s0, s1), (s0, s2), (s1, s0), (s2, s0)\}$ $I = \{s0\}$ $F = (\{s0\}, \{s1\}), (\{s0\}, \{s2\})$

弱公平 Kripke 模型: 弱公平 Kripke 模型的形式与强公平 Kripke 模型相同, 即用 $F \subseteq 2^S \times 2^S$ 取代原来 $K = \langle S, R, I, F \rangle$ 的公平条件。不同的是对公平路径的定义。在这里, 一条路径 $\pi = [s_i]_{i \geq 0}$ 是公平的, 当且仅当对所有 $(f, g) \in F$,

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

按照这样的公平路径的定义, 我们有一个弱公平 Kripke 模型。

弱公平模型的表达能力: 弱公平 Kripke 模型在一些情况下与原公平 Kripke 模型相比较有表达的方便性, 但与原公平 Kripke 模型的表达能力相同。

- 给定公平 Kripke 模型 $K = \langle S, R, I, \{g_1, \dots, g_k\} \rangle$ 。定义 $F' = \{(S, g_1), \dots, (S, g_k)\}$ 。则弱公平 Kripke 模型 $K' = \langle S, R, I, F' \rangle$ 与 K 有相同的公平计算集合。
- 给定弱公平 Kripke 模型 $K = \langle S, R, I, \{(f_1, g_1), \dots, (f_k, g_k)\} \rangle$ 。定义 $F' = \{g_1 \cup (S \setminus f_1), \dots, g_k \cup (S \setminus f_k)\}$ 。则公平 Kripke 模型 $K' = \langle S, R, I, F' \rangle$ 与 K 有相同的公平计算集合。

§2.3 标号Kripke模型

Kripke模型对于系统的描述过于简化，只使用状态集表示性质等。为了表达和计算的简便，可用命题表示性质。我们可将结构中的状态和命题建立联系，使得我们能知道每个状态上哪些命题是成立的。

定义 2.11 给定原子命题的有穷集合 AP 。一个 AP 上的标号Kripke模型是一个四元组 $\langle S, R, I, L \rangle$ 其中 $\langle S, R, I \rangle$ 为Kripke模型， $L: S \rightarrow 2^{AP}$ 为状态集到 AP 的幂集的映射。

$L(s)$ 表示在 s 上成立的所有原子命题组成的集合，即原子命题 p 在 s 上成立当且仅当 $p \in L(s)$ 。假定原子命题集合 AP 和 AP 上的标号Kripke模型 $K = \langle S, R, I, L \rangle$ 为给定。

可达状态： 标号Kripke模型 K 的可达状态即 $K' = \langle S, R, I \rangle$ 的可达状态。

计算与行为： 标号Kripke模型 K 上的路径即 $K' = \langle S, R, I \rangle$ 上的路径。 K 的计算即 K' 的计算，其行为 $[[K]] = [[K']]$ 。

模型语言： 将一个计算上的每个状态替换成这个状态所标的命题集合，我们得到一个命题集合的无穷串。与计算对应的这些无穷串的集合称为模型的语言，定义如下。设 $\pi = [\pi_i]_{i \geq 0}$ 。定义 $L(\pi) = [L(\pi_i)]_{i \geq 0}$ 。集合 $\{L(\pi) \subseteq (2^{AP})^\omega \mid \pi \in [[K]]\}$ 称为 K 的语言，记作 $\mathcal{L}(K)$ 。

可表达性质： 给定 2^{AP} 上的无穷串的集合 $A \subseteq (2^{AP})^\omega$ 。称 A 是标号Kripke模型可表达的当且仅当存在 K 使得 $\mathcal{L}(K) = A$ 。

状态与公式： 记原子命题集合 AP 上所有命题公式的集合为 \mathcal{L}_{AP} 。我们用命题逻辑的公式表示状态的性质。状态 $s \in S$ 满足 $\varphi \in \mathcal{L}_{AP}$ 记作 $K, s \models \varphi$ ，定义如下。

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

定义 $K, X \models \varphi$ ，当且仅当对所有 $s \in X$ 都有 $K, s \models \varphi$ 。

定义 2.12 设 $m = \{x_1, \dots, x_k\} \subseteq AP$ 且 $\{y_1, \dots, y_l\} = AP \setminus m$ 。 $m \models \varphi$ 当且仅当 $(\varphi_{x_1, \dots, x_k}^{1, \dots, 1})_{y_1, \dots, y_l}^{0, \dots, 0}$ 的值为1。

为方便起见，在 K 为给定且不引起混淆的情况下， $K, s \models \varphi$ 和 $K, X \models \varphi$ 亦可分别写做 $s \models \varphi$ 和 $X \models \varphi$ 。

命题 2.32 $s \models \varphi$ 当且仅当 $L(s) \models \varphi$ 。

状态集合与公式的对应： 定义 $[[s]] = (\bigwedge_{p \in L(s)} p) \wedge (\bigwedge_{p \in AP \setminus L(s)} \neg p)$ 。定义 $[[X]] = \bigvee_{s \in X} [[s]]$ 。我们有以下相应性质。

- $X \models \varphi$ 当且仅当 $[[X]] \rightarrow \varphi$ 。
- 若 $X \subseteq Y$ 则 $[[X]] \rightarrow [[Y]]$ 。
- 若 L 为单射函数，则以下成立。 $[[X]] \rightarrow [[Y]]$ 则 $X \subseteq Y$ 。

公式与状态集合的对应： 定义 $[[\varphi]] = \{s \mid s \models \varphi\}$ 。 $[[\varphi]]$ 即满足 φ 的状态的集合。称满足 φ 的状态为 φ 状态。若一条路径上的所有状态都满足 φ ，则称该路径为 φ 路径。若一个计算上的所有状态都满足 φ ，则称该计算为 φ 计算。我们有以下相应性质。

- $\varphi \rightarrow \psi$ 当且仅当 $[[\varphi]] \subseteq [[\psi]]$ 。
- $\varphi \leftrightarrow [[([\varphi])]$ 。
- 若 L 为单射函数，则 $X = [[([\varphi])]$ 。

正确性性质： 我们可将原来用状态集合定义的可达性质、安全性质、可免性质、必达性质重新用公式定义。

- φ 是 $K = \langle S, R, I, L \rangle$ 的可达性质，当且仅当 $[[\varphi]]$ 是 $K' = \langle S, R, I \rangle$ 的可达性质，即 $Rh(I) \cap [[\varphi]] \neq \emptyset$ ，即 $\exists s \in Rh(I). (s \models \varphi)$ 。
- φ 是 $K = \langle S, R, I, L \rangle$ 的安全性质，当且仅当 $[[\varphi]]$ 是 $K' = \langle S, R, I \rangle$ 的安全性质，即 $Rh(I) \subseteq [[\varphi]]$ ，即 $Rh(I) \models \varphi$ 。
- φ 是 $K = \langle S, R, I, L \rangle$ 的可避免性质，当且仅当 $[[\varphi]]$ 是 $K' = \langle S, R, I \rangle$ 的可免性质，即 $\exists B \subseteq S. ((B \cap Rh(I) \neq \emptyset) \wedge (\forall s \in B. (\exists s' \in B. (s \rightarrow s'))) \wedge (\forall s \in B. (s \not\models \varphi)))$ 。
- φ 是 $K = \langle S, R, I, L \rangle$ 的必达性质，当且仅当 $[[\varphi]]$ 是 $K' = \langle S, R, I \rangle$ 的必达性质，即 $\forall B \subseteq S. ((B \cap Rh(I) \neq \emptyset) \wedge (\forall s \in B. (\exists s' \in B. (s \rightarrow s'))) \rightarrow (\exists s \in B. (s \models \varphi)))$ 。

分析方法： 标号 Kripke 模型在原来 Kripke 模型之上建立了状态与命题逻辑公式的联系，原有的概念和分析方法可以经过相应改变后使用。以下仅讨论安全性质与必达性质的推理问题。

安全性质的推理： 定义 $R(\varphi) = [[\{s' \mid s \rightarrow s', s \models \varphi\}]]$ 。若 $R(\varphi) \rightarrow \varphi$ ，则称 φ 为迁移不变。关于安全性质，我们有如下推理规则。设 φ', φ 为公式。

$$\frac{\begin{array}{l} [[I]] \rightarrow \varphi' \\ R(\varphi') \rightarrow \varphi' \\ \varphi' \rightarrow \varphi \end{array}}{\varphi \text{ 是安全性质}}$$

可靠性与完备性： 以上推理规则是可靠的且若 L 为单射函数则推理规则是完备的。假定前提成立，由前两条我们知道 $Rh(I) \models \varphi'$ ，又由第三条我们知道 $Rh(I) \models \varphi$ ，所以 φ 是安全性质，因而推理规则是可靠的。假定 φ 是安全的，取 $\varphi' = [[Rh(I)]]$ ，由 L 为单射函数知 $s \models \varphi'$ 则 $s \in Rh(I)$ ，所以前提条件成立，由此可证 φ 是安全性质，因而推理规则是完备的。

必达性质的推理： 关于必达性质，我们有如下推理规则。设 $\psi_0, \psi_1, \dots, \psi_n, \varphi$ 为公式。

$$\begin{array}{l} [[I]] \rightarrow \psi_0 \\ \forall i \in \{0, \dots, n-1\}. (R(\psi_i \wedge \neg\varphi) \rightarrow \psi_{i+1}) \\ \psi_n \rightarrow \varphi \\ \hline \varphi \text{ 是必达性质} \end{array}$$

可靠性与完备性： 以上推理规则是可靠且对于有穷状态系统若 L 是单射函数则推理规则是完备的。

§2.4 公平标号Kripke模型

我们可以结合标号Kripke模型和公平Kripke模型构造标号公平Kripke模型，这样我们可以使用公式描述性质，也可以对合理的计算设置条件。我们可以在公平Kripke模型加标号也可以在标号Kripke模型加公平条件。本节的选择是在标号Kripke模型上加公平条件。

定义 2.13 给定一个有穷命题集合 AP 。一个 AP 上的公平标号Kripke模型是一个五元组 $\langle S, R, I, L, F \rangle$ 其中 $\langle S, R, I, L \rangle$ 是一个 AP 上的标号Kripke模型， $F \subseteq \mathcal{L}_{AP}$ 为公平性约束的有穷集合。

计算与行为： 公平标号Kripke模型 $K = \langle S, R, I, L, F \rangle$ 的计算就是Kripke模型 $\langle S, R, I \rangle$ 的计算，设 $F = \{\phi_1, \dots, \phi_k\}$ 。 K 的公平计算就是公平Kripke模型 $K' = \langle S, R, I, \{[[\phi_1]], \dots, [[\phi_k]]\} \rangle$ 的公平计算，其行为 $[[K]] = [[K']]$ 。

正确性性质及其分析方法： 由公式与状态集合的对应关系，我们可以相应地定义公平标号Kripke模型的公平安全和公平必达等性质，并将标号Kripke模型和公平Kripke模型的概念和分析方法经过相应改变后使用。以下仅讨论语言及语言非空问题。

模型语言： 将一个计算上的每个状态替换成这个状态所标的命题集合，我们得到一个命题集合的无穷串。与公平计算对应的这些无穷串的集合称为模型的语言，定义如下。集合 $\{L(\pi) \subseteq (2^{AP})^\omega \mid \pi \in [[K]]\}$ 称为 K 的语言，记作 $\mathcal{L}(K)$ 。

语言非空问题： 语言非空问题可转化成模型非空问题。我们有以下结论。

命题 2.33 给定 $K = \langle S, R, I, L, F \rangle$ 且 $F = \{\phi_1, \dots, \phi_k\}$ 。 $\mathcal{L}(K) \neq \emptyset$ ，当且仅当 $[[K]]$ 非空，当且仅当公平Kripke模型 $K' = \langle S, R, I, \{[[\phi_1]], \dots, [[\phi_k]]\} \rangle$ 非空，即 $EmpChecking(K')$ 为 $false$ 。

公平安全性质的推理： 给定 $K = \langle S, R, I, L, F \rangle$ 且 $F = \{\phi_1, \dots, \phi_k\}$ 。设 $\psi_0, \psi_1, \dots, \psi_k, \varphi$ 为公式。我们有以下规则。

$$\frac{\begin{array}{l} \varphi \vee \psi_0 \text{ 是 } \langle S, R, I, L \rangle \text{ 的安全性质} \\ \bigvee_{i=1}^k \psi_i \text{ 是 } \langle S, R, [[\psi_0]], L \rangle \text{ 的必达性质} \\ \forall i \in \{1, \dots, k\}, \neg \phi_i \text{ 是 } \langle S, R, [[\psi_i]], L \rangle \text{ 的安全性质} \end{array}}{\varphi \text{ 是 } K \text{ 的公平安全性质}}$$

可靠性： 假定前提成立。由前提3知，对 $i = 1, \dots, k$ ， ψ_i 状态以及其可达的状态都不是公平状态。由于 ψ_0 不能避开 ψ_1, \dots, ψ_k ，因而 ψ_0 状态不是公平状态。由于 $\varphi \vee \psi_0$ 是 $\langle S, R, I, L \rangle$ 的安全性质，因而所有 K 的公平计算都是 φ 计算，即 φ 是 K 的公平安全性质。由于在以 ψ_0 状态不是公平状态为前提条件的情况下，并不一定保证存在 ψ_1, \dots, ψ_k 使得前提2和前提3成立，因而推理规则不具有完备性。

公平必达性质的推理： 给定 $K = \langle S, R, I, L, F \rangle$ 且 $F = \{\phi_1, \dots, \phi_k\}$ 。设 $\psi_0, \psi_1, \dots, \psi_k, \varphi$ 为公式。我们有以下规则。

$$\frac{\begin{array}{l} \varphi \vee \psi_0 \text{ 是 } \langle S, R, I, L \rangle \text{ 的必达性质} \\ \bigvee_{i=1}^k \psi_i \text{ 是 } \langle S, R, [[\psi_0]], L \rangle \text{ 的必达性质} \\ \forall i \in \{1, \dots, k\}, \neg \phi_i \text{ 是 } \langle S, R, [[\psi_i]], L \rangle \text{ 的安全性质} \end{array}}{\varphi \text{ 是 } K \text{ 的公平必达性质}}$$

可靠性： 假定前提成立，那么 ψ_0 状态不是公平状态。由于 $\varphi \vee \psi_0$ 是 $\langle S, R, I, L \rangle$ 的必达性质，因而所有 K 的公平计算都必然可达 φ 状态，即 φ 是 K 的公平必达性质。

§2.5 例子

例 2.1 (1) 定义 S, R, I 如下。

$\begin{aligned} S &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\ R &= \{(s_0, s_1), (s_0, s_3), (s_1, s_2), (s_1, s_4), (s_2, s_1), (s_3, s_4), (s_4, s_4), (s_4, s_5), (s_5, s_2), (s_5, s_4), (s_5, s_6), (s_6, s_5)\} \\ I &= \{s_0, s_3\} \end{aligned}$

则 $K_1 = (S, R, I)$ 为Kripke模型。

K_1 的可达状态集合为 $\{s_0, s_1, s_2, s_3, s_4\}$ 。

定义 A_1, A_2 如下。

$$\begin{aligned} A_1 &= \{s_0, s_1, s_2, s_3, s_4\} \\ A_2 &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \end{aligned}$$

则 A_1 和 A_2 是模型的安全性质。

根据安全性质分析算法知 $SafetyAnalysis(K_1, A_1)$ 为true且 $SafetyAnalysis(K_1, A_2)$ 为true。

定义 B_1, B_2 如下。

$$\begin{aligned} B_1 &= \{s_2, s_4\} \\ B_2 &= \{s_2, s_4, s_5\} \end{aligned}$$

则 B_1 和 B_2 是模型的必达性质。

根据必达性质分析算法知 $InevitabilityAnalysis(K_1, B_1)$ 为 $true$ 且 $InevitabilityAnalysis(K_1, B_2)$ 为 $true$ 。

(2) 定义 $F = \{\{s_1, s_4\}, \{s_2, s_5\}\}$

则 $K_2 = (S, R, I, F)$ 为公平Kripke模型。

K_2 的可达公平状态集合为 $\{s_0, s_1, s_2\}$ 。

定义 A_3, B_3 如下。

$$A_3 = \{s_0, s_1, s_2\}$$

$$B_3 = \{s_2\}$$

则 A_3 是模型的公平安全性质且 B_3 是模型的公平必达性质。

根据公平安全性质分析算法和公平必达性质分析算法知

$FairSafetyAnalysis(K_2, A_3)$ 为 $true$ 且 $FairInevitabilityAnalysis(K_2, B_3)$ 为 $true$ 。

(3) 设 $AP = \{p, q, r\}$ 。定义 $L : S \rightarrow 2^{AP}$ 如下。

$$\begin{aligned} L(s_0) &= \{p\} & L(s_1) &= \{q\} & L(s_2) &= \{p, q\} & L(s_3) &= \{\} & L(s_4) &= \{r\} \\ qL(s_5) &= \{p, r\} & L(s_6) &= \{q, r\} \end{aligned}$$

则 $K_3 = (S, R, I, L)$ 为 AP 上的标号Kripke模型。

定义 $\varphi_1, \varphi_2, \psi_1, \psi_2$ 如下。

$$\varphi_1 = (p \wedge \neg q \wedge \neg r) \vee (q \wedge \neg r) \vee (\neg p \wedge \neg q)$$

$$\varphi_2 = (\neg p \vee \neg r)$$

$$\psi_1 = (p \wedge q \wedge \neg r) \vee (\neg p \wedge \neg q \wedge r)$$

$$\psi_2 = (p \wedge q \wedge \neg r) \vee (\neg q \wedge r)$$

则以下成立。

$$\begin{aligned} [[\varphi_1]] &= \{s_0, s_1, s_2, s_3, s_4\} & [[\varphi_2]] &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\ [[\psi_1]] &= \{s_2\} & [[\psi_2]] &= \{s_2, s_4, s_5\} \end{aligned}$$

相应地, φ_1 和 φ_2 是模型的安全性质且 ψ_1 和 ψ_2 是模型的必达性质。

(4) 定义 $\Phi = \{\phi_1, \phi_2\}$ 其中 ϕ_1, ϕ_2 定义如下。

$$\phi_1 = (\neg p \wedge (q \leftrightarrow \neg r))$$

$$\phi_2 = (p \wedge (q \leftrightarrow \neg r))$$

则 $K_3 = (S, R, I, L, \Phi)$ 为 AP 上的标号公平Kripke模型。

定义 φ_3, ψ_3 如下。

$$\varphi_3 = (p \vee q) \wedge \neg r$$

$$\psi_3 = (p \wedge q) \wedge \neg r$$

我们有

$$\begin{aligned} [[\phi_1]] &= \{s_1, s_4\} & [[\phi_2]] &= \{s_2, s_5\} \\ [[\varphi_3]] &= \{s_0, s_1, s_2\} & [[\psi_3]] &= \{s_2\} \end{aligned}$$

相应地, φ_3 是模型的公平安全性质且 ψ_3 是模型的公平必达性质。

例 2.2 设 $K_1 = (S, R, I)$ 和 A_1, A_2, B_1, B_2 如例 2.1 中所定义。

(1) A_1 是 K_1 的安全性质的推理证明如下。

定义 $A'_1 = A_1$ 。则以下成立。

$$\begin{aligned} I &\text{ 满足 } A'_1 \\ A'_1 &\text{ 是迁移不变量} \\ A'_1 &\text{ 满足 } A_1 \end{aligned}$$

因而根据推理规则, A_1 是 K_1 的安全性质。

A_2 是 K_1 的安全性质的推理证明如下。

定义 $A'_2 = A_2$ 。则以下成立。

$$\begin{aligned} I &\text{ 满足 } A'_2 \\ A'_2 &\text{ 迁移不变} \\ A'_2 &\text{ 满足 } A_2 \end{aligned}$$

因而根据推理规则, A_2 是 K_1 的安全性质。

注意: 由于 A_2 不是迁移不变, 故不能取 $A'_2 = A_2$ 。

(2) B_1 是 K_1 的必达性质的推理证明如下。

定义 X_i 如下: $X_0 = \{s_0, s_3\}, X_1 = \{s_1, s_4\}, X_2 = \{s_2, s_4\}$ 。

则以下成立。

$$\begin{aligned} I &\text{ 满足 } X_0 \\ \forall i \in \{0, 1\}, R(X_i \setminus B_1) &\text{ 满足 } X_{i+1} \\ X_2 &\text{ 满足 } B_1 \end{aligned}$$

因而根据推理规则, B_1 是 K_1 的必达性质。

类似地可以证明 B_2 是 K_1 的必达性质。

例 2.3 设 $/, \%$ 分别为自然数上的整除和余数函数。设 a, b, x, y, t 为自然数上的函数定义如下。

$$\begin{aligned} a(i) &= i/24 & b(i) &= (i\%24)/8 \\ x(i) &= (i\%8)/4 & y(i) &= (i\%4)/2 \\ t(i) &= i\%2 \end{aligned}$$

定义 $t_1(k, j), \dots, t_8(k, j)$ 如下。

$$\begin{aligned} t_1(k, j) &= a(k) = 0 \wedge a(j) = 1 \wedge b(j) = b(k) \wedge x(j) = x(k) \wedge y(j) = 1 \wedge t(j) = 1 \\ t_2(k, j) &= a(k) = 1 \wedge \neg(x(k) = 0 \vee t(k) = 0) \wedge (j = k) \\ t_3(k, j) &= a(k) = 1 \wedge (x(k) = 0 \vee t(k) = 0) \wedge a(j) = 2 \wedge b(j) = b(k) \wedge (j\%8 = k\%8) \\ t_4(k, j) &= a(k) = 2 \wedge a(j) = a(k) \wedge b(j) = b(k) \wedge x(j) = x(k) \wedge y(j) = 0 \wedge t(j) = t(k) \\ t_5(k, j) &= b(k) = 0 \wedge a(j) = a(k) \wedge b(j) = 1 \wedge x(j) = 1 \wedge y(j) = y(j) \wedge t(j) = 0 \\ t_6(k, j) &= b(k) = 1 \wedge \neg(y(k) = 0 \vee t(k) = 1) \wedge (j = k) \\ t_7(k, j) &= b(k) = 1 \wedge (y(k) = 0 \vee t(k) = 1) \wedge a(j) = a(k) \wedge b(j) = 2 \wedge (j\%8 = k\%8) \\ t_8(k, j) &= b(k) = 2 \wedge a(j) = a(k) \wedge b(j) = b(k) \wedge x(j) = 0 \wedge y(j) = y(k) \wedge t(j) = t(k) \end{aligned}$$

(1) 定义 S, R, I 如下。

$$\begin{array}{l} S = \{s_0, s_1, \dots, s_{71}\} \\ R = \{(s_k, s_j) \mid t_1(k, j) \vee \dots \vee t_8(k, j)\} \\ I = \{s_0, s_1\} \end{array}$$

则 $K_1 = (S, R, I)$ 为Kripke模型。

定义 A_1, B_1 如下。

$$\begin{array}{l} A_1 = \{s_i \mid a(i) \neq 2 \vee b(i) \neq 2\} \\ B_1 = \{s_i \mid a(i) = 2 \vee b(i) = 2\} \end{array}$$

则 A_1 为 K_1 的安全性质, B_1 不是 K_1 的必达性质。

根据安全性质分析算法和必达性质分析算法知

$SafetyAnalysis(K_1, A_1)$ 为true 且 $InevitabilityAnalysis(K_1, B_1)$ 为false。

若该模型用以表示2个进程的互斥算法的运行模型, 则模型具有以上意义的安全(互斥)性质。

设 $R' = R \setminus \{(s, s) \mid s \in S\}$ 且 $K'_1 = (S, R', I)$ 。

则 K'_1 为Kripke模型, A_1 为 K'_1 的安全性质且 B_1 为 K'_1 的必达性质。

根据安全性质分析算法和必达性质分析算法知

$SafetyAnalysis(K'_1, A_1)$ 为true 且 $InevitabilityAnalysis(K'_1, B_1)$ 为true。

(2) 定义 F 为以下集合的集合。

$$\begin{array}{l} \{s_i \mid \neg(a(i) = 0)\} \\ \{s_i \mid \neg(a(i) = 1 \wedge (x(i) = 0 \vee t(i) = 0))\} \\ \{s_i \mid \neg(a(i) = 2)\} \\ \{s_i \mid \neg(b(i) = 0)\} \\ \{s_i \mid \neg(b(i) = 1 \wedge (y(i) = 0 \vee t(i) = 1))\} \\ \{s_i \mid \neg(b(i) = 2)\} \end{array}$$

定义 $K_2 = (S, R, I, F)$ 。则 A_1 为 K_2 的公平安全性质且 B_1 为 K_2 的公平必达性质。

根据公平安全性质分析算法和公平必达性质分析算法知

$FairSafetyAnalysis(K_2, A_1)$ 为true 且 $FairInevitabilityAnalysis(K_2, B_1)$ 为true。

若该模型用以表示2个进程的互斥算法的带有公平约束的运行模型, 则模型具有以上意义的安全和必达性质。

(3) 设 $AP = \{p_1, p_2, p_3, p_4, p_5, p_6\}$ 。定义 $L : S \rightarrow 2^{AP}$ 如下。

$$\begin{array}{l} p_1 \in L(s_i) \quad \text{iff} \quad a(i) = 0 \\ p_2 \in L(s_i) \quad \text{iff} \quad a(i) = 1 \\ p_3 \in L(s_i) \quad \text{iff} \quad a(i) = 2 \\ p_4 \in L(s_i) \quad \text{iff} \quad b(i) = 0 \\ p_5 \in L(s_i) \quad \text{iff} \quad b(i) = 1 \\ p_6 \in L(s_i) \quad \text{iff} \quad b(i) = 2 \end{array}$$

则 $K_3 = (S, R, I, L)$ 为 AP 上的标号 *Kripke* 模型。

定义 φ_1, ψ_1 如下: $\varphi_1 = \neg(p_3 \wedge p_6), \psi_1 = (p_3 \vee p_6)$ 。

则 φ_1 是 K_3 的安全性质, ψ_1 不是 K_3 的必达性质。

定义 $K'_3 = (S, R, I, L)$ 。则 φ_1 是 K'_3 的安全性质且 ψ_1 是 K'_3 的必达性质。

(4) 设 $AP' = AP \cup \{q_1, q_2, q_3, q_4, q_5, q_6\}$ 。定义 $L' : S \rightarrow 2^{AP'}$ 如下。

$$\begin{aligned} p_i \in L'(s_i) & \text{ iff } p_i \in L(s_i) \\ q_1 \in L'(s_i) & \text{ iff } x(i) = 0 \\ q_2 \in L'(s_i) & \text{ iff } x(i) = 1 \\ q_3 \in L'(s_i) & \text{ iff } y(i) = 0 \\ q_4 \in L'(s_i) & \text{ iff } y(i) = 1 \\ q_5 \in L'(s_i) & \text{ iff } t(i) = 0 \\ q_6 \in L'(s_i) & \text{ iff } t(i) = 1 \end{aligned}$$

设 Φ 为以下集合。

$$\{\neg(p_1), \neg(p_2 \wedge (q_1 \vee q_5)), \neg(p_3), \neg(p_4), \neg(p_5 \wedge (q_3 \vee q_6)), \neg(p_6)\}$$

则 $K_4 = (S, R, I, L', \Phi)$ 为 AP' 上的公平标号 *Kripke* 模型, φ_1 为 K_4 的公平安全 (互斥) 性质且 ψ_1 为 K_4 的公平必达 (资源有机会得到使用) 性质。

例 2.4 (1) 设 $K_1 = (S, R, I)$ 和 A_1 如例 2.3 中所定义。 A_1 是 K_1 的安全性质的推理证明如下。

定义 A'_1 为如下集合的并集。

$$\begin{aligned} & \{s_i \mid (a(i) = 0 \wedge b(i) = 0 \wedge x(i) = 0 \wedge y(i) = 0)\} \\ & \{s_i \mid (a(i) = 0 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 0)\} \\ & \{s_i \mid (a(i) = 0 \wedge b(i) = 2 \wedge x(i) = 1 \wedge y(i) = 0)\} \\ & \{s_i \mid (a(i) = 1 \wedge b(i) = 0 \wedge x(i) = 0 \wedge y(i) = 1)\} \\ & \{s_i \mid (a(i) = 1 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 1)\} \\ & \{s_i \mid (a(i) = 1 \wedge b(i) = 2 \wedge x(i) = 1 \wedge y(i) = 1 \wedge t(i) = 1)\} \\ & \{s_i \mid (a(i) = 2 \wedge b(i) = 0 \wedge x(i) = 0 \wedge y(i) = 1)\} \\ & \{s_i \mid (a(i) = 2 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 1 \wedge t(i) = 0)\} \end{aligned}$$

则以下成立。

$$\begin{aligned} & I \text{ 满足 } A'_1 \\ & A'_1 \text{ 是迁移不变量} \\ & A'_1 \text{ 满足 } A_1 \end{aligned}$$

因而根据推理规则, A_1 是 K_1 的安全性质。

(2) 设 $K'_1 = (S, R', I)$ 和 B_1 如例 2.3 中所定义。 B_1 是 K'_1 的必达性质的推理证明如下。

定义 $X_{i,j}$ 为如下。

$$\begin{aligned}
X_{0,1} &= \{s_i \mid (a(i) = 0 \wedge b(i) = 0 \wedge x(i) = 0 \wedge y(i) = 0 \wedge t(i) = 0)\} \\
X_{0,2} &= \{s_i \mid (a(i) = 0 \wedge b(i) = 0 \wedge x(i) = 0 \wedge y(i) = 0 \wedge t(i) = 1)\} \\
X_{1,1} &= \{s_i \mid (a(i) = 1 \wedge b(i) = 0 \wedge x(i) = 0 \wedge y(i) = 1 \wedge t(i) = 1)\} \\
X_{1,2} &= \{s_i \mid (a(i) = 0 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 0 \wedge t(i) = 0)\} \\
X_{2,1} &= \{s_i \mid (a(i) = 1 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 0 \wedge t(i) = 0)\} \\
X_{2,2} &= \{s_i \mid (a(i) = 1 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 1 \wedge t(i) = 1)\} \\
X_{3,1} &= \{s_i \mid (a(i) = 2 \wedge b(i) = 1 \wedge x(i) = 1 \wedge y(i) = 1 \wedge t(i) = 0)\} \\
X_{3,2} &= \{s_i \mid (a(i) = 1 \wedge b(i) = 2 \wedge x(i) = 1 \wedge y(i) = 1 \wedge t(i) = 1)\}
\end{aligned}$$

定义 $X_0 = X_{0,1} \cup X_{0,2}$, $X_1 = X_{1,1} \cup X_{1,2}$, $X_2 = X_{2,1} \cup X_{2,2} \cup A_2$, $X_3 = X_{3,1} \cup X_{3,2}$ 。则以下成立。

$$\begin{aligned}
&I \text{ 满足 } X_0 \\
&\forall i \in \{0, 1, 2\}, R'(X_i \setminus A_2) \text{ 满足 } X_{i+1} \\
&X_3 \text{ 满足 } B_1
\end{aligned}$$

因而根据推理规则, B_1 是 K'_1 的必达性质。

§2.6 练习

1. 给定Kripke模型 $K = \langle S, R, I \rangle$ 和 $A \subseteq S$ 。判断以下说法的正确性： A 是可避免性质，当且仅当 K 有一条由 I 出发可达非 A 非平凡强连通分量的非 A 路径。
2. 给定Kripke模型 $K = \langle S, R, I \rangle$ 和 $B, A \subseteq S$ 。结合安全与可达和必达概念，我们可定义系统的安全可达和安全必达性质如下： B 为 A 安全可达性质当且仅当存在可达 B 的计算且该计算中第一个 B 状态前的所有状态都是 A 状态； B 为 A 安全必达性质当且仅当 B 必达且在所有计算中第一个 B 状态前的所有状态都是 A 状态。设计算法分别检查 B 是否为 A 安全可达性质和 B 是否为 A 安全必达性质。
3. 给定Kripke模型 $K = \langle S, R, I \rangle$ 和 $B, A \subseteq S$ 。定义 (B, A) 路径为满足以下条件的路径：至少一个 B 状态出现在该路径中且同时或之后有 A 状态出现。设计基于不动点计算的算法以检查 K 中是否存在初始状态为起点的 (B, A) 路径。
4. 给定公平Kripke模型 $K = \langle S, R, I, F \rangle$ 和 $A \subseteq S$ 。通过对模型进行改造，用模型非空问题算法检查 A 是否是 K 的可达性质。
5. 给定公平Kripke模型 $K = \langle S, R, I, F \rangle$ 和 $A \subseteq S$ 。通过对模型进行改造，用模型非空问题算法检查 A 是否是 K 的可避免性质。
6. 设 $S, R, R', I, AP', L', \varphi_1, \psi_1$ 如例2.3中所定义。则 $K = (S, R, I, L')$ 和 $K' = (S, R', I, L')$ 为 AP' 上的标号Kripke模型。用推理方法证明 φ_1 是 K 的安全性质且 ψ_1 是 K' 的必达性质。

§3 基于一阶逻辑的迁移系统

本章介绍基于一阶逻辑的以变量赋值为特点的模型。这类模型将状态分解为不同的分量，状态的迁移基于变量的重新赋值。有些子类型的模型亦包含控制状态及控制状态的转移规则。

一阶逻辑： 给定常元和函数符号集合 F ，谓词符号集合 P 。

$B = (F, P)$ 上一阶逻辑公式集合记为 WFF_B 。

B 上的项的集合记为 T_B 。

WFF_B 中不带量词的公式集合记为 QFF_B 。

B 的解释确定了迁移模型描述中常元符号、函数符号和谓词符号的含义。

给定 B 的解释 I 。一个赋值 σ 满足公式 φ 当且仅当 $I(\varphi)(\sigma) = true$ ，记作 $\sigma \models_I \varphi$ 。所有赋值都满足公式 φ ，记作 $\models_I \varphi$ 。

变量限制在 V 中的项的集合记作 $T_{(B,V)}$ 。

自由变量限制在 V 中的公式的集合记作 $WFF_{(B,V)}$ 。

自由变量限制在 V 中的 QFF_B 公式的集合记作 $QFF_{(B,V)}$ 。

以下假定 $B = (F, P)$ 以及 B 的解释 $I = (D, I_0)$ 为给定。

假定 P 中包含命题常量 $true$ 和 $false$ 。为方便起见，用 $\varphi = \emptyset$ 表示 $\models_I (\varphi \leftrightarrow false)$ 。

不动点公式： 设 $g : WFF_B \rightarrow WFF_B$ 。定义 $g^0(\varphi) = \varphi$ 和 $g^{i+1}(\varphi) = g^i(\varphi)$ 。

若 $\forall k \geq 0. (g^k(false) \rightarrow \psi)$ 且 $\forall \psi'. (\forall k \geq 0. (g^k(false) \rightarrow \psi') \rightarrow (\psi \rightarrow \psi'))$ ，则称 ψ 为 $g(Z)$ 的最小不动点公式，记作 $\mu Z.g(Z)$ 。若 $\forall k \geq 0. (\psi \rightarrow g^k(true))$ 且 $\forall \psi'. (\forall k \geq 0. (\psi' \rightarrow g^k(true)) \rightarrow (\psi' \rightarrow \psi))$ ，则称 ψ 为 $g(Z)$ 的最大不动点公式，记作 $\nu Z.g(Z)$ 。

称一个不动点公式为良定义的，当且仅当存在一个具有所述相关性质的公式。

若对任意公式 φ, ψ 都有 $\varphi \rightarrow \psi$ 蕴涵 $g(\varphi) \rightarrow g(\psi)$ ，则称函数 g 为单调函数。

为论述方便，以下假定我们使用的一阶逻辑具有所需的表达能力且单调函数的不动点公式都是良定义的。

§3.1 卫式迁移模型

卫式迁移模型(Guarded Command Transition Systems)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，卫式迁移模型包含初始条件的描述。在一阶逻辑的基础上，增加以下符号集合：

$$\{\rightarrow, :=\}$$

给定变量集合 V 。定义 (B, V) 上的迁移如下。

$$\varphi \rightarrow (x_1, x_2, \dots, x_k) := (e_1, e_2, \dots, e_k)$$

其中 $\varphi \in QFF_{(B,V)}$ ， $\{x_1, \dots, x_k\} \subseteq V$ 且各不相同， $\{e_1, \dots, e_k\} \subseteq T_{(B,V)}$ 。一个迁移表示模型的一个原子动作。

定义 3.1 一个 (B, V) 上的卫式迁移模型是一个二元组 (T, Θ) 其中 T 为 (B, V) 上迁移的有穷集合， $\Theta \in QFF_{(B,V)}$ 为初始条件。

以下假定卫式迁移模型 $M = (T, \Theta)$ 为给定。

状态与状态空间： 变量的赋值 σ 代表了计算过程中变量在某一时刻的状态。因此称 σ 为状态。模型的状态空间为 V 中变量取值的组合 $\Sigma = \{\sigma \mid \sigma(x) \in D, x \in V\}$ 。状态的有效部分取决于 V 中变量的取值。若 $V = \{v_1, \dots, v_n\}$ ，则 σ 亦可表示为 n 元组 $\langle \sigma(v_1), \dots, \sigma(v_n) \rangle$ 。若 $\sigma \models_I \varphi$ ，则称 σ 为 φ 状态。 φ 状态的集合记为 $[[\varphi]]$ 。

可执行迁移： 设 t 是 $\varphi \rightarrow (x_1, x_2, \dots, x_k) := (e_1, e_2, \dots, e_k)$ 为一个迁移。 t 在状态 σ 可执行当且仅当 $\sigma \models_I \varphi$ 。

死锁状态： 若在状态 σ 没有可执行迁移，则称 σ 为死锁状态。

迁移关系： 定义 $\sigma \xrightarrow{t} \sigma'$ 当且仅当 t 在状态 σ 可执行且 $\sigma' = \sigma[x_1/I(e_1)(\sigma)] \cdots [x_k/I(e_k)(\sigma)]$ 。模型的迁移关系 \rightarrow 为 Σ^2 的一个子集。 $\sigma \xrightarrow{M} \sigma'$ 当且仅当以下一种情况成立。

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

$\sigma \xrightarrow{M} \sigma'$ 在不引起混淆时简写为 $\sigma \rightarrow \sigma'$ 。

可达状态： 记由 φ 状态可达的状态集合为 $Rh(\varphi)$ 。我们有 $Rh(\varphi) = \{\sigma' \mid \sigma \xrightarrow{*} \sigma', \sigma \models_I \varphi\}$ 。 M 的可达状态集合即由 Θ 状态可达的状态集合 $Rh(\Theta) = \{\sigma' \mid \sigma \xrightarrow{*} \sigma', \sigma \models_I \Theta\}$ 。

路径： 无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的路径，当且仅当 $[\sigma_i]_{i \geq 0}$ 是状态集 Σ 上的一个无穷序列且对任意 $i \geq 0$ 都有 $\sigma_i \rightarrow \sigma_{i+1}$ 。

计算与行为： 无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的计算，当且仅当 $[\sigma_i]_{i \geq 0}$ 是 M 的路径且 $\sigma_0 \models_I \Theta$ 。 M 的计算的集合称为 M 的行为，记作 $[[M]]$ 。

§3.1.1 正确性性质

定义 3.2 (安全性质) φ 是 M 的安全性质，当且仅当 M 的所有计算都是 φ 计算。

定义 3.3 (必达性质) φ 是 M 的必达性质，当且仅当 M 的所有计算都能到达 φ 状态。

设 $T = \{p_i \rightarrow e_i \mid i = 1, \dots, k\}$ 。

记 $\bigvee_{i=1}^k p_i$ 为 $E(T)$ 。记 $T \cup \{E(T) \rightarrow (x_1) := (x_1)\}$ 为 T^+ 。

定义 3.4 设 W 为迁移集合。设 t 为 $p \rightarrow (x_1, x_2, \dots, x_k) := (e_1, e_2, \dots, e_k)$ 。定义 $[W]\psi$ 如下。

$$\begin{aligned} [{}]\psi &= true \\ [t]\psi &= p \rightarrow \psi(e_1/x_1, \dots, e_n/x_n) \\ [X \cup Y]\psi &= [X]\psi \wedge [Y]\psi \end{aligned}$$

命题 3.1 $\varphi \equiv [T^+]\psi$ 当且仅当 $\forall \sigma. (I(\varphi)\sigma \rightarrow \forall \sigma'. ((\sigma \rightarrow \sigma') \rightarrow I(\psi)\sigma'))$ 。

定义 $\langle W \rangle \varphi = \neg [W]\neg \varphi$ 。

命题 3.2 φ 是 M 的安全性质，当且仅当 $\forall k \geq 0. (\Theta \wedge (\langle T^+ \rangle)^k \neg \varphi = \emptyset)$ 。

命题 3.3 φ 是 M 的必达性质当且仅当 $\forall \psi. ((\psi \wedge \Theta \neq \emptyset) \wedge (\psi \rightarrow \langle T^+ \rangle \psi) \rightarrow (\psi \wedge \varphi \neq \emptyset))$ 。

§3.1.2 卫式迁移模型与标号Kripke模型的等价

给定 (B, V) 上的卫式迁移模型 $M = (T, \Theta)$ 。给定 $AP = \{p_1, \dots, p_n\}$ 上的标号Kripke模型 $K = \langle S, R, I, L \rangle$ 。

设 $BP \subseteq QFF_{(B, V)}$ 。

定义 \mathcal{L}_{BP} 为以 BP 中的命题为原子命题的布尔公式。

设 $\zeta : AP \rightarrow BP$ 是一一对应关系。

定义 $\zeta^{-1}(\varphi)$ 为对所有 φ 中出现的 BP 中的公式用 ζ 关系中对应的 AP 中的命题替换得到的结果。

定义 3.5 M 与 K 是 ζ 计算等价的，当且仅当对任意 M 的计算 $[\sigma_i]_{i \geq 0}$ 存在 K 的计算 $[s_i]_{i \geq 0}$ 满足 $\forall i \geq 0. \forall \psi \in BP. (\sigma_i \models \psi \leftrightarrow s_i \models \zeta^{-1}(\psi))$ 且对任意 K 的计算 $[s_i]_{i \geq 0}$ 存在 M 的计算 $[\sigma_i]_{i \geq 0}$ 满足 $\forall i \geq 0. \forall p \in AP. (s_i \models p \leftrightarrow \sigma_i \models \zeta(p))$ 。

定义 3.6 M 与 K 是 ζ 互模拟等价的，当且仅当存在对称关系 $\alpha \subseteq \Sigma \times S$ 使得以下成立。

- 对任意初始状态 $\sigma \in \Sigma$ 存在初始状态 $s \in S$ 使得 $(\sigma, s) \in \alpha$ ，反之亦然；
- 对任意 $p \in AP$ ，若 $(\sigma, s) \in \alpha$ 则 $s_i \models p$ 当且仅当 $\sigma_i \models \zeta(p)$ ；
- 对任意 $(\sigma, s) \in \alpha$ ，若 $(\sigma \rightarrow \sigma')$ 则存在 $s' \in S$ 使得 $(\sigma', s') \in \alpha$ ，反之亦然。

命题 3.4 若 M 与 K 为 ζ 互模拟等价，则 M 与 K 为 ζ 计算等价。

命题 3.5 设 M 与 K 是 ζ 计算等价的。设 $\varphi \in \mathcal{L}_{BP}$ 。则 φ 是 M 的安全性质当且仅当 $\zeta^{-1}(\varphi)$ 是 K 的安全性质； φ 是 M 的必达性质当且仅当 $\zeta^{-1}(\varphi)$ 是 K 的必达性质。

卫式迁移模型的标号Kripke模型的构造： 我们可以构造与卫式迁移模型等价的标号Kripke模型，用标号Kripke模型性质的分析方法来分析卫式迁移模型的性质。

给定 (B, V) 上的卫式迁移模型 $M = (T, \Theta)$ 其中 $V = \{v_1, \dots, v_n\}$ 。

定义

$$\begin{aligned} S &= D^n. \\ R &= \{(\langle \sigma(v_1), \dots, \sigma(v_n) \rangle, \langle \sigma'(v_1), \dots, \sigma'(v_n) \rangle) \mid \sigma \rightarrow \sigma'\}. \\ I &= \{(\langle \sigma(v_1), \dots, \sigma(v_n) \rangle) \mid \sigma \models_I \Theta\}. \end{aligned}$$

设 $BP = \{\psi_1, \dots, \psi_n\} \subseteq QFF_{(B, V)}$ 。

定义 $AP = \{p_1, \dots, p_n\}$ 且定义 $L : S \rightarrow 2^{AP}$ 如下： $p_i \in L(\langle \sigma(v_1), \dots, \sigma(v_n) \rangle) \leftrightarrow \sigma \models \psi_i$ 。

定义 $\zeta : AP \rightarrow BP$ 如下： $\zeta(p_1) = \psi_1, \dots, \zeta(p_n) = \psi_n$ 。

定义 $(M)^{km} = \langle S, R, I, L \rangle$ 。

命题 3.6 M 与 $(M)^{km}$ 是 ζ 计算等价的。

推论 3.1 设 $\varphi \in \mathcal{L}_{BP}$ 。则 φ 是 M 的安全性质当且仅当 $\zeta^{-1}(\varphi)$ 是 $(M)^{km}$ 的安全性质； φ 是 M 的必达性质当且仅当 $\zeta^{-1}(\varphi)$ 是 $(M)^{km}$ 的必达性质。

§3.2 公平卫式迁移模型

我们可以在卫式迁移模型的基础上设置公平计算的条件，定义公平卫式迁移模型。

定义 3.7 给定变量集合 V 。一个 (B, V) 上的公平卫式迁移模型是一个三元组 (T, Θ, Φ) 其中 (T, Θ) 为 (B, V) 上的卫式迁移模型， $\Phi \subseteq QFF_{(B, V)}$ 为公平条件，是一个自由变量在 V 中且不带量词的公式的有穷集合。

以下假定公平卫式迁移模型 $M = (T, \Theta, \Phi)$ 为给定。

状态空间与迁移关系： M 的状态空间等同于 (T, Θ) 的状态空间。 M 的状态迁移关系等同于 (T, Θ) 的状态迁移关系。

公平路径： 无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的路径，当且仅当 $[\sigma_i]_{i \geq 0}$ 是 (T, Θ) 的路径。无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的公平路径，当且仅当 $[\sigma_i]_{i \geq 0}$ 是 M 的路径且对所有 $\phi \in \Phi$ 都有无限多个 i 使得 $\sigma_i \models \phi$ 。

公平计算与行为： 无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的计算，当且仅当 $[\sigma_i]_{i \geq 0}$ 是 (T, Θ) 的计算。无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的公平计算，当且仅当 $[\sigma_i]_{i \geq 0}$ 是 M 的计算且 $[\sigma_i]_{i \geq 0}$ 是 M 的公平路径。 M 的公平计算的集合称为 M 的行为，记作 $[[M]]$ 。

公平状态： 一个状态是公平状态，当且仅当存在一条从该状态出发的公平路径。定义 $\Sigma_F = \nu Z. \bigwedge_{f \in F} \mu Y. ((f \wedge \langle T^+ \rangle(Z)) \vee \langle T^+ \rangle(Y))$ 。

命题 3.7 σ 是公平状态当且仅当 $\sigma \models \Sigma_F$ 。

§3.2.1 正确性性质

定义 3.8 (公平安全性质) φ 是 M 的公平安全性质，当且仅当 M 的所有公平计算都是 φ 计算。

定义 3.9 (公平必达性质) φ 是 M 的公平必达性质，当且仅当 M 的所有公平计算都能到达 φ 状态。

命题 3.8 φ 是 M 的公平安全性质，当且仅当 $\Theta \wedge \mu Y. ((\neg \varphi \wedge \Sigma_F) \vee \langle T^+ \rangle(Y)) = \emptyset$ 。

命题 3.9 ψ 是 K 的公平必达性质当且仅当 $\Theta \wedge \nu Z. (\neg \varphi \wedge \bigwedge_{f \in F} \langle T^+ \rangle (\mu Y. ((Z \wedge f) \vee (\neg \varphi \wedge \langle T^+ \rangle(Y)))))) = \emptyset$ 。

§3.2.2 公平卫式迁移模型与公平标号 Kripke 模型的等价

给定 (B, V) 上的公平卫式迁移模型 $M = (T, \Theta, \Phi)$ 。给定 $AP = \{p_1, \dots, p_n\}$ 上的公平标号 Kripke 模型 $K = \langle S, R, I, L, F \rangle$ 。

设 $BP \subseteq QFF_{(B, V)}$ 。

设 $\zeta : AP \rightarrow BP$ 是一一对应关系。

定义 3.10 M 与 K 是 ζ 计算等价的，当且仅当对任意 M 的公平计算 $[\sigma_i]_{i \geq 0}$ 存在 K 的公平计算 $[s_i]_{i \geq 0}$ 满足 $\forall i \geq 0. \forall \psi \in BP. (\sigma_i \models \psi \leftrightarrow s_i \models \zeta^{-1}(\psi))$ 且当且仅当对任意 K 的公平计算 $[s_i]_{i \geq 0}$ 存在 M 的公平计算 $[\sigma_i]_{i \geq 0}$ 满足 $\forall i \geq 0. \forall p \in AP. (s_i \models p \leftrightarrow \sigma_i \models \zeta(p))$ 。

命题 3.10 设 M 与 K 是 ζ 计算等价的。设 $\varphi \in \mathcal{L}_{BP}$ 。则 φ 是 M 的公平安全性质当且仅当 $\zeta^{-1}(\varphi)$ 是 K 的公平安全性质； φ 是 M 的公平必达性质当且仅当 $\zeta^{-1}(\varphi)$ 是 K 的公平必达性质。

公平卫式迁移模型的公平标号Kripke模型的构造： 类似于卫式迁移模型，我们可以构造与公平卫式迁移模型等价的公平标号Kripke模型，用公平标号Kripke模型性质的分析方法来分析公平卫式迁移模型的性质。

给定 (B, V) 上的卫式迁移模型 $M = (T, \Theta, \Phi)$ 其中 $V = \{v_1, \dots, v_n\}$ 。

设 $BP = \{\psi_1, \dots, \psi_n\} \subseteq QFF_{(B, V)}$ 且 $\forall \phi \in \Phi. (\phi \in \mathcal{L}_{BP})$ 。

$\langle S, R, I \rangle$ 的定义同其在§3.1.2中的定义。

定义 $AP = \{p_1, \dots, p_n\}$ 且定义 $L : S \rightarrow 2^{AP}$ 如下： $p_i \in L(\langle \sigma(v_1), \dots, \sigma(v_n) \rangle) \leftrightarrow \sigma \models \psi_i$ 。

定义 $\zeta : AP \rightarrow BP$ 如下： $\zeta(p_1) = \psi_1, \dots, \zeta(p_n) = \psi_n$ 。

定义 $F = \{\zeta^{-1}(\phi) \mid \phi \in \Phi\}$ 。

定义 $(M)^{fkm} = \langle S, R, I, L, F \rangle$ 。

命题 3.11 M 与 $(M)^{fkm}$ 是 ζ 计算等价的。

推论 3.2 设 $\varphi \in \mathcal{L}_{BP}$ 。则 φ 是 M 的公平安全性质当且仅当 $\zeta^{-1}(\varphi)$ 是 $(M)^{fkm}$ 的公平安全性质； φ 是 M 的公平必达性质当且仅当 $\zeta^{-1}(\varphi)$ 是 $(M)^{fkm}$ 的公平必达性质。

§3.3 谓词公式迁移模型

本节介绍一个更为一般的离散迁移模型，称为谓词公式迁移模型，模型中没有显式的变量赋值，代替变量赋值的是用以描述变量值变化关系的谓词公式。谓词公式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。给定变量集合 V 。定义 $V' = \{v' \mid v \in V\}$ 。

定义 3.11 一个 (B, V) 上的谓词公式迁移模型是一个二元组 (ρ, Θ) 其中 $\rho \in QFF_{(B, V \cup V')}$ 为迁移关系，是一个自由变量在 $V \cup V'$ 中且不带量词的公式； $\Theta \in QFF_{(B, V)}$ 为初始条件，是一个自由变量在 V 中且不带量词的公式。

以下假定谓词公式迁移模型 $M = (\rho, \Theta)$ 为给定。

状态与状态空间 变量的赋值 σ 代表了计算过程中变量在某一时刻的状态。因此称 σ 为状态。模型的状态空间为 V 中变量取值的组合 $\Sigma = \{\sigma \mid \sigma(x) \in D, x \in V\}$ 。状态的有效部分取决于 V 中变量的取值。若 $V = \{v_1, \dots, v_n\}$ ，则 σ 亦可表示为 n 元组 $\langle \sigma(v_1), \dots, \sigma(v_n) \rangle$ 。

可执行迁移： 迁移 ρ 在状态 σ 可执行，当且仅当存在 σ' 使得 $\sigma[v'_1/\sigma'(v_1)] \cdots [v'_n/\sigma'(v_n)] \models_I \rho$ 。

迁移关系： 定义 $\sigma \xrightarrow{\rho} \sigma'$ 当且仅当 $\sigma[v'_1/\sigma'(v_1)] \cdots [v'_n/\sigma'(v_n)] \models_I \rho$ 。模型的迁移关系 \rightarrow 为 Σ^2 的一个子集。 $\sigma \xrightarrow{M} \sigma'$ 当且仅当 $\sigma \xrightarrow{\rho} \sigma'$ 或 ρ 在状态 σ 不可执行且 $\sigma' = \sigma$ 。 $\sigma \xrightarrow{M} \sigma'$ 在不引起混淆时简写为 $\sigma \rightarrow \sigma'$ 。

可达状态： 记由 φ 状态可达的状态集合为 $Rh(\varphi)$ 。我们有 $Rh(\varphi) = \{\sigma' \mid \sigma \xrightarrow{*} \sigma', \sigma \models_I \varphi\}$ 。 M 的可达状态集合即由 Θ 状态可达的状态集合 $Rh(\Theta) = \{\sigma' \mid \sigma \xrightarrow{*} \sigma', \sigma \models_I \Theta\}$ 。

路径： 无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的路径，当且仅当 $[\sigma_i]_{i \geq 0}$ 是状态集 Σ 上的一个无穷序列且对任意 $i \geq 0$ 都有 $\sigma_i \rightarrow \sigma_{i+1}$ 。

计算与行为： 无穷序列 $[\sigma_i]_{i \geq 0}$ 称为 M 的计算，当且仅当 $[\sigma_i]_{i \geq 0}$ 是 M 的路径且 $\sigma_0 \models_I \Theta$ 。 M 的计算的集合称为 M 的行为，记作 $[[M]]$ 。

正确性性质 正确性性质的讨论类似于卫式迁移模型，这里不再详述。

卫式迁移模型与谓词公式迁移模型的等价 给定 (B, V) 上的卫式迁移模型 $M = (T, \Theta)$ 。设 $V = \{v_1, \dots, v_n\}$ 。设 t 为 $p \rightarrow (x_1, \dots, x_m) = (e_1, \dots, e_m)$ 。定义 $tr(t)$ 如下。

$$tr(t) = (p \wedge \bigwedge_{i=1}^m (x'_i \leftrightarrow e_i) \wedge \bigwedge_{y \in V \setminus \{x_1, \dots, x_m\}} (y' \leftrightarrow y))$$

定义 $\rho = (\bigvee_{t \in T} tr(t))$ 。则 $M' = (\rho, \Theta)$ 为谓词公式迁移模型且以下命题成立。

命题 3.12 $[[M']] = [[M]]$ 。

公平谓词公式迁移模型 公平计算的讨论类似于卫式迁移模型，即我们可以在谓词公式迁移模型的基础上设置公平计算的条件以定义公平谓词公式迁移模型，这里不再详述。

§3.4 流程图模型

流程图模型(Flowchart Models)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。一个指令表示模型的一个原子动作。在一阶逻辑的基础上，我们增加以下两个集合的符号：

$$\begin{aligned} \text{辅助符号集 } H &= \{:=, :, \text{goto}, \text{if}, \text{else}, \text{fi}\} \\ \text{标号符号集 } LB &= \{\text{beg}, \text{end}, l_1, l_2, \dots\} \end{aligned}$$

给定变量集合 V 。定义 (B, V) 上的流程图模型的两指令如下。

$$\begin{aligned} & \overline{l_1: (x_1, \dots, x_k) := (e_1, \dots, e_k) \text{ goto } l_2} \\ & \overline{l_1: \text{if } (b) \text{ goto } l_2 \text{ else goto } l_3 \text{ fi}} \end{aligned}$$

其中 $l_1, l_2, l_3 \in LB$ 且 $l_1 \neq \text{end}$ ， $\{x_1, \dots, x_k\} \subseteq V$ 且各不相同， $\{e_1, \dots, e_k\} \subseteq T_{(B, V)}$ ， $b \in QFF_{(B, V)}$ 。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。

定义 3.12 一个 (B, V) 上的流程图模型 $M = T$ 其中 T 为满足以下条件的指令集合。

$$\begin{aligned} & \overline{\text{标号 } \text{beg} \text{ 必须有定义}} \\ & \overline{\text{标号 } \text{end} \text{ 不被定义}} \\ & \overline{\text{出现在标号定义中的除 } \text{end} \text{ 外的标号必须有定义}} \\ & \overline{\text{每个标号最多被定义一次}} \end{aligned}$$

以下假定流程图模型 $M = (T)$ 为给定。

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

$$\frac{\begin{array}{l} t \text{ 为 } l_i: (x_1, \dots, x_k) := (e_1, \dots, e_k) \text{ goto } l_{i+1}, \\ \text{且 } \sigma_{i+1} = \sigma_i[x_1/I(e_1)(\sigma_i)] \cdots [x_k/I(e_k)(\sigma_i)] \end{array}}{\begin{array}{l} t \text{ 为 } l_i: (\text{if } (b) \text{ goto } l' \text{ else goto } l'' \text{ fi}, \\ \sigma_{i+1} = \sigma_i \text{ 且 } \sigma_i \models b \text{ 则 } l_{i+1} = l', \text{ 否则 } l_{i+1} = l'' \end{array}}$$

模型的迁移关系 \xrightarrow{M} 为 $(LB \times \Sigma)^2$ 的一个子集。 $(l_i, \sigma_i) \xrightarrow{M} (l_{i+1}, \sigma_{i+1})$ 当且仅当以下一种情况成立。

$$\frac{\begin{array}{l} \text{存在指令 } t \in T \text{ 使得 } (l_i, \sigma_i) \xrightarrow{t} (l_{i+1}, \sigma_{i+1}) \text{ 或} \\ \text{不存在定义标号 } l_i \text{ 的指令且 } (l_i, \sigma_i) = (l_{i+1}, \sigma_{i+1}) \end{array}}{\quad}$$

$(l, \sigma) \xrightarrow{M} (l', \sigma')$ 在不引起混淆时简写为 $(l, \sigma) \rightarrow (l', \sigma')$ 。流程图模型的迁移有如下特点。

命题 3.13 若 $(l, \sigma) \rightarrow (l', \sigma')$ 且 $(l, \sigma) \rightarrow (l'', \sigma'')$ 则 $(l', \sigma') = (l'', \sigma'')$ 。

计算： M 的一个计算是状态集 $LB \times \Sigma$ 上的一个无穷序列 $[(l_i, \sigma_i)]_{i \geq 0}$ 满足 $l_0 = beg$ 且对任意 $i \geq 0$ 都有 $(l_i, \sigma_i) \rightarrow (l_{i+1}, \sigma_{i+1})$ 。

可达状态： 以 σ 为起点的计算可到达的状态的集合为

$$\{(l, \sigma') \mid (beg, \sigma) \xrightarrow{*} (l, \sigma')\}$$

M 的可达状态集合为 $\{(l, \sigma') \mid (beg, \sigma) \xrightarrow{*} (l, \sigma'), \sigma \in \Sigma\}$ 。

终止状态： M 在初始变量状态为 σ 时计算能够终止，当且仅当存在 σ' 使得 $(beg, \sigma) \xrightarrow{*} (end, \sigma')$ 。若存在 σ 使得 $(beg, \sigma) \xrightarrow{*} (end, \sigma')$ ，则称 σ' 为以 σ 为起点的计算的终止状态。

计算结果的唯一性： 流程图模型的计算结果有唯一性，表述如下。

推论 3.3 给定流程图模型 M 和状态 σ 。若 $(beg, \sigma) \xrightarrow{*} (end, \sigma')$ 且 $(beg, \sigma) \xrightarrow{*} (end, \sigma'')$ 则 $\sigma' = \sigma''$ 。

正确性性质： 设 $A \subseteq LB \times \Sigma$ 。

定义 3.13 (安全性质) A 是 M 的安全性质，当且仅当 M 的所有计算都是 A 计算。

定义 3.14 (必达性质) A 是 M 的必达性质，当且仅当 M 的所有计算都能到达 A 状态。

§3.4.1 正确性问题

流程图模型的基本正确性问题包括模型计算的终止性以及模型的终止状态和初始状态的关系。 M 的前置条件是对 M 的初始状态的变量状态的限制条件。 M 的后置条件是对 M 的终止状态的变量状态的限制条件。前置条件和后置条件可以是谓词或公式。设 φ 和 ψ 是谓词。

- 如果模型初始时的变量状态满足条件 φ ，给定 ψ ，如果计算终止，则计算终止时的变量状态满足 ψ 。这个要求称为模型对于前置条件 φ 和后置条件 ψ 的部分正确性。
- 如果模型初始时的变量状态满足条件 φ ，给定 ψ ，则要求计算能够终止且计算终止时的变量状态满足 ψ 。这个要求称为模型对于前置条件 φ 和后置条件 ψ 的完全正确性。

完全正确性包括了模型的终止性，即如果模型初始时的变量状态满足条件 φ ，则要求计算能够终止。

终止性： 设 φ 是谓词。 M 对于前置条件 φ 的终止性定义如下。

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

命题 3.14 M 对于前置条件 $true$ 是终止的，当且仅当 $\{(end, \sigma) \mid \sigma \in \Sigma\}$ 是 M 的必达性质。

部分正确性： 设 φ 和 ψ 是谓词。 M 对于前置条件 φ 和后置条件 ψ 的部分正确性定义如下。

$$\text{部分正确性: } \forall \sigma. (\varphi(\sigma) \rightarrow \forall \sigma'. (((beg, \sigma) \xrightarrow{*} (end, \sigma')) \rightarrow \psi(\sigma')))$$

命题 3.15 M 对于前置条件 $true$ 和后置条件 ψ 是部分正确的，当且仅当 $\{(end, \sigma) \mid \psi(\sigma)\} \cup \{(l, \sigma) \mid l \neq end\}$ 是 M 的安全性质。

完全正确性： 设 φ 和 ψ 是谓词。 M 对于前置条件 φ 和后置条件 ψ 的完全正确性定义如下。

$$\text{完全正确性: } \forall \sigma. (\varphi(\sigma) \rightarrow \exists \sigma'. (((beg, \sigma) \xrightarrow{*} (end, \sigma')) \wedge \psi(\sigma')))$$

命题 3.16 M 对于前置条件 φ 和后置条件 ψ 是完全正确的，当且仅当 M 对于前置条件 φ 和后置条件 ψ 是部分正确的且 M 对于前置条件 φ 是终止的。

以公式为前置条件和后置条件的正确性问题： 相应地可以讨论以公式为前置条件和后置条件的正确性问题。设 φ 和 ψ 为公式。我们可以把公式通过 I 解释成为谓词，则有类似的正确性问题的描述。 M 对于前置条件 φ 的终止性和 M 对于前置条件 φ 和后置条件 ψ 的部分正确性和完全正确性定义如下。

$$\text{终止性: } \forall \sigma. (I(\varphi)(\sigma) \rightarrow \exists \sigma'. ((beg, \sigma) \xrightarrow{*} (end, \sigma')))$$

$$\text{部分正确性: } \forall \sigma. (I(\varphi)(\sigma) \rightarrow \forall \sigma'. (((beg, \sigma) \xrightarrow{*} (end, \sigma')) \rightarrow I(\psi)(\sigma')))$$

$$\text{完全正确性: } \forall \sigma. (I(\varphi)(\sigma) \rightarrow \exists \sigma'. (((beg, \sigma) \xrightarrow{*} (end, \sigma')) \wedge I(\psi)(\sigma')))$$

M 对于前置条件 φ 和后置条件 ψ 的部分正确性和完全正确性分别记作 $\models_I \{\varphi\}M\{\psi\}$ 和 $\models_I [\varphi]M[\psi]$ 。由完全正确性的定义知 M 对于前置条件 φ 的终止性为 $\models_I [\varphi]M[true]$ 。

正确性与安全性质和必达性质的关系 以谓词 φ 为模型 T 的初始状态满足的条件（前置断言）等价于将 T 的初始状态集限制在 $\{(beg, \sigma) \mid \varphi(\sigma)\}$ 之内。在这个受限的模型下，终止性可以表达为必达性质(Inevitability)，即 $\{(end, \sigma) \mid \sigma \in \Sigma\}$ 是否必然可达。部分正确性（以 ψ 为后置断言）可以表达为安全性质(Safety)，即是否所有可达状态都满足 $\{(end, \sigma) \mid \psi(\sigma)\} \cup \{(L, \sigma) \mid L \neq end, \sigma \in \Sigma\}$ 。

§3.4.2 流程图模型与卫式迁移模型的等价

给定 (B, V) 上的流程图模型 M 和 B 的解释 $I = (D, I_0)$ 。

设 (B', V') 是 (B, V) 的扩充。给定 (B', V') 上的卫式迁移模型 $M' = (T', \Theta)$ 和 B' 的解释 $I' = (D, I'_0)$ 。

给定函数 $\zeta : LB \rightarrow WFF_{B'}$ 。

定义 3.15 M 与 M' 是 ζ 计算等价的，当且仅当对任意 M 的计算 $[(l_i, \sigma_i)]_{i \geq 0}$ 存在 M' 的计算 $[\sigma'_i]_{i \geq 0}$ 满足 $\forall i \geq 0. ((\forall \varphi. (\sigma_i \models_I \varphi \leftrightarrow \sigma'_i \models_{I'} \varphi)) \wedge \sigma'_i \models_{I'} \zeta(l_i))$ 且对任意 M' 的计算 $[\sigma'_i]_{i \geq 0}$ 存在 M 的计算 $[(l_i, \sigma_i)]_{i \geq 0}$ 满足 $\forall i \geq 0. ((\forall \varphi. (\sigma_i \models_I \varphi \leftrightarrow \sigma'_i \models_{I'} \varphi)) \wedge \sigma'_i \models_{I'} \zeta(l_i))$ 。

命题 3.17 设 M 与 $M' = (T', \Theta)$ 是 ζ 计算等价的。设 φ, ψ 为公式。 M 对于前置条件 φ 和后置条件 ψ 是部分正确的当且仅当 $\zeta(end) \rightarrow \psi$ 是 $M'' = (T', \Theta \wedge \varphi)$ 的安全性质； M 对于前置条件 φ 是终止的当且仅当 $\zeta(end)$ 是 $M'' = (T', \Theta \wedge \varphi)$ 的必达性质。

流程图模型的卫式迁移模型的构造： 给定 (B, V) 上的流程图模型 $M = (T)$ 和 B 的解释 $I = (D, I_0)$ 。

定义 $B' = (F \cup LB, P)$ 和 $V' = V \cup \{pc\}$ 。

定义 I'_0 为 I_0 的扩充，其扩充部分将 LB 中的符号解释到 D 中不同的元素。

设 N 为流程图模型指令的子集。

用 N 构造卫式迁移模型的迁移集合的算法 $g_1(N)$ ，定义如下。

若	N	$=$	$\{l : (x_1, \dots, x_k) := (e_1, \dots, e_k)\} \cup N_1,$
则	$g_1(N)$	$=$	$\{pc = l \rightarrow (x_1, \dots, x_k, pc) := (e_1, \dots, e_k, l')\} \cup g_1(N_1);$
若	N	$=$	$\{l : (if (b) goto l' else goto l'' fi)\} \cup N_1,$
则	$g_1(N)$	$=$	$\{pc = l \wedge b \rightarrow (pc) := (l'), pc = l \wedge \neg b \rightarrow (pc) := (l'')\} \cup g_1(N_1);$
若	N	$=$	$\{\},$
则	$g_1(N)$	$=$	$\{\}.$

定义 $M^* = (g_1(T), pc = beg)$ 。则 M^* 为卫式迁移模型。

命题 3.18 定义 $\zeta(l)$ 为 $(pc = l)$ 。流程图模型 $M = (T)$ 与卫式迁移模型 $M^* = (g_1(T), pc = beg)$ 是 ζ 计算等价的。

推论 3.4 设 φ, ψ 为公式。 M 对于前置条件 φ 和后置条件 ψ 是部分正确的当且仅当 $pc = end \rightarrow \psi$ 是卫式迁移模型 $M' = (g_1(T), pc = beg \wedge \varphi)$ 的安全性质； M 对于前置条件 φ 是终止的当且仅当 $pc = end$ 是卫式迁移模型 M' 的必达性质。

§3.4.3 并发流程图模型

可以用多个流程图模型组成并发流程图模型。并发流程图模型的计算以其各分量交错计算的形式进行。并发流程图模型使用共享变量互相影响各分量的计算。

定义 3.16 一个 (B, V) 上的并发流程图模型表示为 $M = (M_1, \dots, M_n)$ 其中 M_1, \dots, M_n 为流程图模型。

设 M_1, \dots, M_n 和 $M = (M_1, \dots, M_n)$ 为给定。

状态与状态空间： 并发流程图模型的状态有两个部分：即各分量的标号和变量状态。模型的状态空间为各分量的标号和 V 中变量取值的组合 $LB^n \times \Sigma$ 。

状态迁移关系： 模型的迁移关系 \xrightarrow{M} 为 $(LB^n \times \Sigma)^2$ 的一个子集。 $(l_1, \dots, l_j, \dots, l_n, \sigma) \xrightarrow{M} (l_1, \dots, l'_j, \dots, l_n, \sigma')$ 当且仅当以下一种情况成立。

$$\begin{array}{l} \text{存在指令 } t \in M_j \text{ 使得 } (l_j, \sigma) \xrightarrow{t} (l'_j, \sigma') \text{ 或} \\ \text{对于所有 } 1 \leq k \leq n, M_k \text{ 中不存在定义标号 } l_k \text{ 的指令, 且 } (l_j, \sigma) = (l'_j, \sigma') \end{array}$$

计算： M 的一个计算是状态集 $LB^n \times \Sigma$ 上的一个无穷序列 $[(l_{1,i}, \dots, l_{n,i}, \sigma_i)]_{i \geq 0}$ 满足 $\forall 0 \leq k \leq n. (l_{k,0} = \text{beg})$ 且对任意 $i \geq 0$ 都有 $(l_{1,i}, \dots, l_{n,i}, \sigma_i) \rightarrow (l_{1,i+1}, \dots, l_{n,i+1}, \sigma_{i+1})$ 。

可达状态： 以 σ 为起点的计算可到达的状态的集合为

$$\{(l_1, \dots, l_n, \sigma') \mid (\text{beg}, \dots, \text{beg}, \sigma) \xrightarrow{*} (l_1, \dots, l_n, \sigma')\}$$

M 的可达状态集合为 $\{(l_1, \dots, l_n, \sigma') \mid (\text{beg}, \dots, \text{beg}, \sigma) \xrightarrow{*} (l_1, \dots, l_n, \sigma'), \sigma \in \Sigma\}$ 。

终止状态： 若存在 σ 使得 $(\text{beg}, \dots, \text{beg}, \sigma) \xrightarrow{*} (\text{end}, \dots, \text{end}, \sigma')$ ，则称 σ' 为以 σ 为起点的计算的一个终止状态。由于计算的不确定性，有两种类型的终止性，即存在起点为 σ 的计算的终止状态和对所有起点为 σ 的计算都能够终止。

并发流程图模型与卫式迁移模型的等价 并发流程图模型与卫式迁移模型的计算等价与单个流程图模型与卫式迁移模型的计算等价类似，其区别仅在于并发流程图的标号表示为 n 元组而单个流程图模型的仅有一个标号，参照定义3.15。给定 (B, V) 上的并发流程图模型 $M = (M_1, \dots, M_n)$ 和 B 的解释 $I = (D, I_0)$ 。定义 $B' = (F \cup LB, P)$ 和 $V' = V \cup \{pc_1, \dots, pc_n\}$ 。定义 I'_0 为 I_0 的扩充，其扩充部分将 LB 中的符号解释到 D 中不同的元素。分别构造 M_i （其中 $0 \leq i \leq n$ ）的 ζ_i 计算等价的 $(B', V \cup \{pc_i\})$ 上的卫式迁移模型 $M'_i = (T'_i, \Theta_i)$ 。令 $M' = (T'_1 \cup \dots \cup T'_n, \Theta_1 \wedge \dots \wedge \Theta_n)$ 且 $\zeta = (\zeta_1, \dots, \zeta_n)$ 。则 M' 与 M 为 ζ 计算等价。

§3.5 结构化程序模型

结构化程序模型(While-Programs)是一阶逻辑的扩充。结构化程序模型的主要要素有赋值、顺序复合、条件语句和循环语句。在一阶逻辑的基础上,我们增加以下集合的符号:

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

定义 3.17 给定变量集合 V 。一个 (B, V) 上的结构化程序模型是一个字符串,其集合 $\mathcal{L}_{\circ}^{(B, V)}$ 定义如下。

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

其中 $x \in V$ 为变量, $e \in T_{(B, V)}$ 为项, $b \in QFF_{(B, V)}$ 为公式。

为书写方便,在不引起歧义的情况下,模型结尾的 ϵ 可省略不写。

状态与状态空间: 结构化程序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 $\Sigma = \{\sigma \mid \sigma(x) \in D, x \in V\}$ 。系统的状态空间为模型和 V 中变量取值的组合 $\mathcal{L}_{\circ}^{(B, V)} \times \Sigma$ 。

状态迁移关系: 系统状态的迁移关系 \rightarrow 为 $(\mathcal{L}_{\circ}^{(B, V)} \times \Sigma)^2$ 的子集。 $(S_0, \sigma_0) \rightarrow (S_1, \sigma_1)$ 当且仅当以下一项成立。

S_0	条件	S_1	σ_1
$x := t; S$		S	$\sigma_0[x/I(t)(\sigma_0)]$
$if\ (b)\ then\ T\ else\ T'\ fi; S$	$\sigma_0 \models_I b$	$T; S$	σ_0
$if\ (b)\ then\ T\ else\ T'\ fi; S$	$\sigma_0 \not\models_I b$	$T'; S$	σ_0
$while\ b\ do\ T\ od; S$	$\sigma_0 \models_I b$	$T; while\ (b)\ do\ T\ od; S$	σ_0
$while\ b\ do\ T\ od; S$	$\sigma_0 \not\models_I b$	S	σ_0
ϵ		ϵ	σ_0

结构化程序模型的迁移有如下特点。

命题 3.19 若 $(S, \sigma) \rightarrow (S', \sigma')$ 且 $(S, \sigma) \rightarrow (S'', \sigma'')$ 则 $(S', \sigma') = (S'', \sigma'')$ 。

计算: 结构化程序模型 M 的一个计算是状态集 $\mathcal{L}_{\circ}^{(B, V)} \times \Sigma$ 上的一个无穷序列 $[(S_i, \sigma_i)]_{i \geq 0}$ 满足 $S_0 = M$ 且对任意 $i \geq 0$ 都有 $(S_i, \sigma_i) \rightarrow (S_{i+1}, \sigma_{i+1})$ 。

可达状态: M 在初始变量状态为 σ 时可到达的状态的集合为 $\{(M', \sigma') \mid (M, \sigma) \xrightarrow{*} (M', \sigma')\}$ 。 M 在的可达状态集合为 $\{(M', \sigma') \mid (M, \sigma) \xrightarrow{*} (M', \sigma'), \sigma \in \Sigma\}$ 。

终止状态： M 在初始变量状态为 σ 时计算能够终止，当且仅当存在 σ' 使得 $(M, \sigma) \xrightarrow{*} (\epsilon, \sigma')$ 。若存在 σ 使得 $(M, \sigma) \xrightarrow{*} (\epsilon, \sigma')$ ，则称 σ' 为 M 的一个终止状态。

计算结果的唯一性： 结构化程序模型的计算结果有唯一性，表述如下。

推论 3.5 给定结构化程序模型 S 和状态 σ 。若 $(S, \sigma) \xrightarrow{*} (\epsilon, \sigma')$ 且 $(S, \sigma) \xrightarrow{*} (\epsilon, \sigma'')$ 则 $\sigma' = \sigma''$ 。

正确性性质： 设 $A \subseteq \mathcal{L}_{\circ}^{(B, V)} \times \Sigma$ 。

定义 3.18 (安全性质) A 是 M 的安全性质，当且仅当 M 的所有计算都是 A 计算。

定义 3.19 (必达性质) A 是 M 的必达性质，当且仅当 M 的所有计算都能到达 A 状态。

§3.5.1 正确性问题

类似于流程图模型，结构化程序模型的基本正确性问题包括模型计算的终止性以及模型的终止状态和初始状态的关系。

终止性： 设 φ 是谓词。 M 对于前置条件 φ 的终止性定义如下。

$$\text{终止性: } \forall \sigma. (\varphi(\sigma) \rightarrow \exists \sigma'. ((M, \sigma) \xrightarrow{*} (\epsilon, \sigma')))$$

命题 3.20 M 对于前置条件 $true$ 是终止的，当且仅当 $\{(\epsilon, \sigma) \mid \sigma \in \Sigma\}$ 是 M 的必达性质。

部分正确性： 设 φ 和 ψ 是谓词。 M 对于前置条件 φ 和后置条件 ψ 的部分正确性定义如下。

$$\text{部分正确性: } \forall \sigma. (\varphi(\sigma) \rightarrow \forall \sigma'. (((M, \sigma) \xrightarrow{*} (\epsilon, \sigma')) \rightarrow \psi(\sigma')))$$

命题 3.21 M 对于前置条件 $true$ 和后置条件 ψ 是部分正确的，当且仅当 $\{(\epsilon, \sigma) \mid \psi(\sigma)\} \cup \{(T, \sigma) \mid T \neq \epsilon\}$ 是 M 的安全性质。

完全正确性： 设 φ 和 ψ 是谓词。 M 对于前置条件 φ 和后置条件 ψ 的完全正确性定义如下。

$$\text{完全正确性: } \forall \sigma. (\varphi(\sigma) \rightarrow \exists \sigma'. (((M, \sigma) \xrightarrow{*} (\epsilon, \sigma')) \wedge \psi(\sigma')))$$

命题 3.22 M 对于前置条件 φ 和后置条件 ψ 是完全正确的，当且仅当 M 对于前置条件 φ 和后置条件 ψ 是部分正确的且 M 对于前置条件 φ 是终止的。

以公式为前置条件和后置条件的正确性问题： 相应地可以定义以公式为前置条件和后置条件的正确性问题。设 φ 和 ψ 为公式。 M 对于前置条件 φ 的终止性和 M 对于前置条件 φ 和后置条件 ψ 的部分正确性和完全正确性定义如下。

$$\text{终止性: } \forall \sigma. (I(\varphi)(\sigma) \rightarrow \exists \sigma'. ((M, \sigma) \xrightarrow{*} (\epsilon, \sigma')))$$

$$\text{部分正确性: } \forall \sigma. (I(\varphi)(\sigma) \rightarrow \forall \sigma'. (((M, \sigma) \xrightarrow{*} (\epsilon, \sigma')) \rightarrow I(\psi)(\sigma')))$$

$$\text{完全正确性: } \forall \sigma. (I(\varphi)(\sigma) \rightarrow \exists \sigma'. (((M, \sigma) \xrightarrow{*} (\epsilon, \sigma')) \wedge I(\psi)(\sigma')))$$

M 对于前置条件 φ 和后置条件 ψ 的部分正确性和完全正确性分别记作 $\models_I \{\varphi\}M\{\psi\}$ 和 $\models_I [\varphi]M[\psi]$ 。由完全正确性可得出 M 对于前置条件 φ 的终止性为 $\models_I [\varphi]M[true]$ 。

正确性与安全性质和必达性质的关系 以谓词 φ 为模型 S 的初始状态满足的条件（前置断言）等价于将 S 的初始状态限制在 $\{(S, \sigma) \mid \varphi(\sigma)\}$ 之内。在这个受限的模型下，终止性可以表达为必达性质(Inevitability)，即 $\{(\epsilon, \sigma) \mid \sigma \in \Sigma\}$ 是否必然可达。部分正确性（以 ψ 为后断置言）可以表达为安全性质(Safety)，即是否所有可达状态都满足 $\{(\epsilon, \sigma) \mid \psi(\sigma)\} \cup \{(S', \sigma) \mid S' \neq \epsilon, \sigma \in \Sigma\}$ 。

§3.5.2 结构化程序模型与流程图模型的等价

给定 (B, V) 上的结构化程序模型 M 和流程图模型 M' 。

定义 3.20 结构化程序模型 M 与流程图模型 M' 是计算等价的，当且仅当存在一一对应关系 $\zeta : LB \rightarrow \mathcal{L}_{\circlearrowleft}^{(B, V)}$ 满足 $\zeta(beg) = M$ 且 $\zeta(end) = \epsilon$ 使得对任意 M 的计算 $[(S_i, \sigma_i)]_{i \geq 0}$ 存在 M' 的计算 $[(\zeta^{-1}(S_i), \sigma_i)]_{i \geq 0}$ 且对任意 M' 的计算 $[(l_i, \sigma_i)]_{i \geq 0}$ 存在 M 的计算 $[(\zeta(l_i), \sigma_i)]_{i \geq 0}$ 。

命题 3.23 设结构化程序模型 M 与流程图模型 M' 是计算等价的。设 φ, ψ 为公式。 M 对于前置条件 φ 和后置条件 ψ 是部分正确的当且仅当 M' 对于 φ 和 ψ 是部分正确的， M 对于前置条件 φ 是终止的当且仅当 M' 对于 φ 是终止的。

结构化程序模型的流程图模型的构造： 给定 (B, V) 上的结构化程序模型 M 和 B 的解释 $I = (D, I_0)$ 。

对 M 中的每个语句 T 的入口标记一个唯一标号 $lb(T)$ 满足 $lb(M) = beg$ 且 $lb(\epsilon) = end$ 且同一语句若出现在不同地方则赋予不同标号。记打上标号的 M 为 M_{lb} 。

我们用带标号的结构化程序模型 M_{lb} 构造流程图模型。构造算法 $g_2(S)$ 其输入 S 为 M_{lb} 的片段，定义如下。

S	$g_2(S)$
$l : \epsilon$	$\{\}$
$l : x := t; S_1$	$\{l : (x) := (t) \text{ goto } lb(S_1)\} \cup g_2(S_1)$
$l : \text{if } b \text{ then } T_1 \text{ else } T_2 \text{ fi}; S_1$	$\{l : \text{if } (b) \text{ goto } lb(T_1) \text{ else goto } lb(T_2) \text{ fi}\} \cup g_2(T_1; S_1) \cup g_2(T_2; S_1)$
$l : \text{while } b \text{ do } T_1 \text{ od}; S_1$	$\{l : \text{if } (b) \text{ goto } lb(T_1) \text{ else goto } lb(S_1) \text{ fi}\} \cup g_2(T_1; l : \epsilon)$

定义 $M^* = g_2(M_{lb})$ 。则 M^* 为流程图模型。

命题 3.24 结构化程序模型 M 与流程图模型 $M^* = g_2(M_{lb})$ 是计算等价的。

推论 3.6 设 φ, ψ 为公式。 M 对于前置条件 φ 和后置条件 ψ 是部分正确的当且仅当 $M^* = g_2(M_{lb})$ 对于 φ 和 ψ 是部分正确的， M 对于前置条件 φ 是终止的当且仅当 M^* 对于 φ 是终止的。

§3.6 例子

例 3.1 设 $B = (F, P)$ 其中 $F = \{ncr, wait, cr, 0, 1\}$ 且 $P = \{=\}$ 。设 $V = \{a, b, x, y, t\}$ 。

(1) 定义 $M_1 = (T, \Theta)$ 如下。

$$\Theta = (a = ncr \wedge b = ncr \wedge x = 0 \wedge y = 0)。$$

T 为以下迁移的集合。

$$\begin{aligned} a = ncr & \longrightarrow (a, y, t) := (wait, 1, 1) \\ a = wait \wedge \neg(x = 0 \vee t = 0) & \longrightarrow (a) := (wait) \\ a = wait \wedge (x = 0 \vee t = 0) & \longrightarrow (a) := (cr) \\ a = cr & \longrightarrow (a, y) := (ncr, 0) \\ b = ncr & \longrightarrow (b, x, t) := (wait, 1, 0) \\ b = wait \wedge \neg(y = 0 \vee t = 1) & \longrightarrow (b) := (wait) \\ b = wait \wedge (y = 0 \vee t = 1) & \longrightarrow (b) := (cr) \\ b = cr & \longrightarrow (b, x) := (ncr, 0) \end{aligned}$$

则 M_1 为 (B, V) 上的卫式迁移模型。

定义 φ_1, φ_2 和 B 的解释 $I = (D, I_0)$ 如下。

$$\varphi_1 = \neg(a = ncr \wedge b = ncr)。$$

$$\varphi_2 = (a = ncr \vee b = ncr)。$$

$$D = \{0, 1, 2\}。$$

I_0 定义如下。

$$I_0(ncr) = I_0(0) = 0$$

$$I_0(wait) = I_0(1) = 1$$

$$I_0(cr) = 2$$

$$I_0(=) = =$$

则 φ_1 是 M_1 的安全性质， φ_2 不是 M_1 的必达性质，

若该模型用以表示2个进程的互斥算法的运行模型，则模型具有以上意义的安全性质。

将 T 中的8个迁移根据其排列的顺序分别命名为 t_1, \dots, t_8 。

设 $M'_1 = (T \setminus \{t_2, t_6\}, \Theta)$ 。

则 M'_1 为 (B, V) 上的卫式迁移模型， φ_1 为 M'_1 的安全性质且 φ_2 为 M'_1 的必达性质。

设 AP 和 $K_3 = (S, R, I, L)$ 如例2.3所定义。

设 $BP = \{a = ncr, a = wait, a = cr, b = ncr, b = wait, b = cr\}$ 。

定义 $\zeta : AP \rightarrow BP$ 如下。

$$\zeta(p_1) = a = ncr$$

$$\zeta(p_2) = a = wait$$

$$\zeta(p_3) = a = cr$$

$$\zeta(p_4) = b = ncr$$

$$\zeta(p_5) = b = wait$$

$$\zeta(p_6) = b = cr$$

则 M_1 与 K_3 为 ζ 互模拟等价。

由此可得 φ_1 为 M_1 的安全性质当且仅当 $\neg(p_1 \wedge p_4)$ 为 K_3 的安全性质， φ_2 为 M_1 的必达性质当且仅当 $(p_1 \vee p_4)$ 为 K_3 的必达性质。

(2) 设 Φ 为以下公式的集合。

$$\begin{aligned} &\neg(a = ncr) \\ &\neg(a = wait \wedge (x = 0 \vee t = 0)) \\ &\neg(a = cr) \\ &\neg(b = ncr) \\ &\neg(b = wait \wedge (y = 0 \vee t = 1)) \\ &\neg(b = cr) \end{aligned}$$

设 $M_2 = (T, \Theta, \Phi)$ 。

则 M_2 为 (B, V) 上的公平卫式迁移模型。

φ_1 为 M_2 的公平安全性质且 φ_2 为 M_2 的公平必达性质。

若该模型用以表示2个进程的互斥算法的带有公平约束的运行模型，则模型具有以上意义的安全和必达性质。

设 AP' 和 $K_4 = (S, R, I, L', \Phi)$ 如例2.3所定义。

设 $BP' = BP \cup \{x = 0, x = 1, y = 0, y = 1, t = 0, t = 1\}$ 。

定义 $\zeta' : AP' \rightarrow BP'$ 为 $zeta$ 的扩充且其扩充部分定义如下。

$$\begin{aligned} \zeta(p_7) &= x = 0 \\ \zeta(p_8) &= x = 1 \\ \zeta(p_9) &= y = 0 \\ \zeta(p_{10}) &= y = 1 \\ \zeta(p_{11}) &= t = 0 \\ \zeta(p_{12}) &= t = 1 \end{aligned}$$

则 M_4 中的 Φ 可由 M_2 中的 Φ 应用 $(\zeta')^{-1}$ 映射得到，进而可得 M_2 与 K_4 为 ζ' 计算等价。

由此可得 φ_1 为 M_2 的公平安全性质当且仅当 $\neg(p_1 \wedge p_4)$ 为 K_4 的公平安全性质， φ_2 为 M_2 的公平必达性质当且仅当 $(p_1 \vee p_4)$ 为 K_4 的公平必达性质。

例 3.2 设 $B = (F, P)$ 其中 $F = \{0, 1, +, fib\}$ 且 $P = \{<, \leq\}$ 。设 $V = \{y_1, y_2, z, r\}$ 。定义 T 为以下指令的集合。

$$\begin{aligned} beg &: (y_1, y_2, r) := (1, 1, 1); goto s_1 \\ s_1 &: if (y_2 \leq z) goto s_2 else goto end \\ s_2 &: (y_1, y_2) := (y_2, y_1 + y_2); goto s_3 \\ s_3 &: (r) := (r + 1); goto s_1 \end{aligned}$$

则 $M_3 = T$ 为 (B, V) 上的流程图模型。

设 B 的解释 $I = (D, I_0)$ 定义如下。

- D 为自然数集合。

- I_0 定义如下。

$$\begin{aligned}
I_0(0) &= 0 \\
I_0(1) &= 1 \\
I_0(2) &= 2 \\
I_0(+) &= + \\
I_0(>) &= < \\
I_0(\leq) &= \leq \\
I_0(fib) &= fib
\end{aligned}$$

其中 fib 为斐波那契函数, 定义如下 $fib(1) = fib(2) = 1$, 对 $i \geq 2$, $fib(i) = fib(i-1) + fib(i-2)$ 。
定义 φ_1 如下。

$$\varphi_1 = fib(r) \leq z \wedge z < fib(r+1)$$

M_3 所需满足的正确性性质为以 $z > 0$ 为前断言 φ_1 为后断言的完全正确性。

该模型用以表示一个在整数加减法运算上计算满足 $fib(r) \leq z$ 的最大的 r (记为 $r = bif(z)$), 则该模型具有这个意义下的完全正确性。

设 $B' = (F \cup \{beg, s_1, s_2, s_3, end\}, P)$ 和 $V' = V \cup \{pc\}$ 。

设 I'_0 为 I_0 的扩充, 其扩充部分将 $\{beg, s_1, s_2, s_3, end\}$ 中的符号解释到 D 中不同的元素。

定义卫式迁移模型 $M'_3 = (T', \Theta)$ 其中 Θ 为 $pc = beg$ 且 T' 为以下迁移的结合。

$$\begin{aligned}
pc = beg &\rightarrow (y_1, y_2, r, pc) := (1, 1, 1, s_1) \\
pc = s_1 \wedge (y_2 \leq z) &\rightarrow (pc) := (s_2) \\
pc = s_1 \wedge \neg(y_2 \leq z) &\rightarrow (pc) := (end) \\
pc = s_2 &\rightarrow (y_1, y_2, pc) := (y_2, y_1 + y_2, s_3) \\
pc = s_3 &\rightarrow (r, pc) := (r + 1, s_1)
\end{aligned}$$

定义 $\zeta(l)$ 为 $(pc = l)$ 。则模型 M_3 与 M'_3 为 ζ 计算等价。

由此可得 M_3 满足以 $z > 0$ 为前断言 φ_1 为后断言的完全正确性当且仅当 $(pc = end) \rightarrow \varphi_1$ 是 M'_3 的安全性质且 $(pc = end)$ 是 M'_3 的必达性质。

定义 $M''_3 = (T', \Theta')$ 其中 $\Theta' = \Theta \wedge z > 0$ 。则 M_3 满足以 $z > 0$ 为前断言 φ_1 为后断言的完全正确性当且仅当 $(pc = end) \rightarrow \varphi_1$ 是 M''_3 的安全性质且 $(pc = end)$ 是 M''_3 的必达性质。

例 3.3 设 (B, V) 如例 3.2 所定义。定义 S 为以下字符串。

```

y1 := 1; y2 := 1; r := 1;
while (y2 ≤ z) do y0 := y1; y1 := y2; y2 := y0 + y1; r := r + 1 od;
ε

```

则 $M_4 = S$ 为 $(B, V \cup \{y_0\})$ 上的结构化程序模型。

设 B 的解释 $I = (D, I_0)$ 如例 3.2 所定义。

定义 φ_1 如下 (同例 3.2 中定义)。

$$\varphi_1 = fib(r) \leq z \wedge z < fib(r+1)$$

M_4 所需满足的正确性性质为以 $z > 0$ 为前断言 φ_1 为后断言的完全正确性。

该模型用以表示一个在整数加减法运算上计算 $r = bif(z)$ 的程序, 则该模型具有这个意义下的完全正确性。

设 $M'_4 = T$ 为 (B, V) 上的流程图模型其中 T 为以下指令的集合。

```

beg : (y1) := (1); goto s11
s11 : (y2) := (1); goto s12
s12 : (y3) := (1); goto s1
s1 : if (y2 ≤ z) goto s21 else goto end
s21 : (y0) := (y1); goto s22
s22 : (y1) := (y2); goto s2
s2 : (y2) := (y0 + y1); goto s3
s3 : (r) := (r + 1); goto s1

```

则模型 M_4 与 M'_4 计算等价且见证等价相应的 ζ 如下。

$$\begin{aligned}
 \zeta(\textit{beg}) &= S \\
 \zeta(s_{11}) &= S_{11} \equiv y_2 := 1; S_{12} \\
 \zeta(s_{12}) &= S_{12} \equiv r := 1; S_1 \\
 \zeta(s_1) &= S_1 \equiv \textit{while} (y_2 \leq z) \textit{ do } y_0 := y_1; y_1 := y_2; y_2 := y_0 + y_1 \textit{ od}; S_4 \\
 \zeta(s_{21}) &= S_{21} \equiv y_0 := y_1; S_{22} \\
 \zeta(s_{22}) &= S_{21} \equiv y_1 := y_2; S_2 \\
 \zeta(s_2) &= S_2 \equiv y_2 := y_0 + y_1; S_3 \\
 \zeta(s_3) &= S_3 \equiv r := r + 1; S_{12} \\
 \zeta(\textit{end}) &= S_4 \equiv \epsilon
 \end{aligned}$$

由此可得 M_4 满足以 $z > 0$ 为前断言 φ_1 为后断言的完全正确性当且仅当 M'_4 满足以 $z > 0$ 为前断言 φ_1 为后断言的完全正确性。

§3.7 练习

1. 设计3个进程的互斥协议的(公平)卫式迁移模型, 说明模型中所用到的各类符号及其解释, 定义安全性质和必达性质并分析其正确性。
2. 设计建立在自然数加减法基础上计算斐波那契数的流程图模型和结构化程序模型。分别说明这两个模型中所用到的各类符号及其解释并定义其所需要满足的正确性性质。
3. 设计建立在自然数加减法基础上计算最大公约数的流程图模型和结构化程序模型。分别说明这两个模型中所用到的各类符号及其解释并定义其所需要满足的正确性性质。
4. 定义一种具备一定合理性的流程图模型与标号Kripke模型的计算等价关系, 并给出一种能够满足前述等价关系的、由流程图模型构造标号Kripke模型的方法。
5. 定义一种具备一定合理性的结构化程序模型与标号Kripke模型的计算等价关系, 并给出一种能够满足前述等价关系的、由结构化程序模型构造标号Kripke模型的方法。

§4 基于迁移标号的迁移系统

本章介绍以标号的迁移为特点的模型。这类模型除了与Kripke模型类似的基本元素外，迁移有自己的标号，便于描述一个计算是通过什么样的迁移或什么类型的迁移进行的。

§4.1 标号迁移系统

定义 4.1 一个标号迁移系统是一个四元组 (Σ, S, Δ, I) 其中 Σ 为标号的有穷集合, S 为有穷状态集合, $\Delta \subseteq S \times \Sigma \times S$ 为标号迁移关系, $I \subseteq S$ 为初始状态集。

以下假定标号迁移系统 $M = (\Sigma, S, \Delta, I)$ 为给定。

无穷字符串: 一个 Σ 上的无穷字符串是 Σ 上的无穷序列。无穷序列 $w = [a_i]_{i \geq 1}$ 是 Σ 上的无穷字符串当且仅当对于所有 $i \geq 1$, $a_i \in \Sigma$ 。

无穷字符串上的运行: 用 $s \xrightarrow{a} s'$ 表示存在从 s 到 s' 的 a 迁移, 即 $(s, a, s') \in \Delta$ 。给定一个 Σ 上的无穷字符串 $w = [a_i]_{i \geq 1}$ 。状态集 S 上的无穷序列 $\pi = [s_i]_{i \geq 0}$ 是 w 上的一个运行, 当且仅当对所有 $i \geq 0$ 有 $s_i \xrightarrow{a_{i+1}} s_{i+1}$ 。

运行: 状态集 S 上的无穷序列 π 是 M 的一个运行, 当且仅当存在 Σ 上的无穷字符串 w 使得 π 是 w 上的一个运行。

运行上的无穷字符串: 给定一个 Σ 上的无穷字符串 $w = [a_i]_{i \geq 1}$ 。 w 是运行 $\pi = [s_i]_{i \geq 0}$ 上的无穷字符串, 当且仅当对所有 $i \geq 0$ 有 $s_i \xrightarrow{a_{i+1}} s_{i+1}$ 。

确定型与非确定型系统: 我们可以将标号迁移系统分为确定型系统和非确定型系统。确定型系统的初始状态只有一个且其迁移关系满足以下限制:

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

命题 4.1 给定一个无穷字符串 w 。对确定型系统, 若 π 和 π' 是 w 上的运行, 则 $\pi = \pi'$ 。

双标号迁移系统: 与标号Kripke结构类似, 我们可以在标号迁移系统的状态上添加一些信息, 构造双标号迁移系统。设 AP 为给定命题集合。

定义 4.2 一个 AP 上的双标号迁移系统是一个五元组 $(\Sigma, S, \Delta, I, L)$ 其中 (Σ, S, Δ, I) 为标号迁移系统, $L: S \rightarrow 2^{AP}$ 为状态标号函数。

§4.2 无穷字符串上的自动机

对于标号迁移系统而言, 运行就是满足迁移关系的起点为初始状态的无穷序列。为了增加表达能力, 更精确地描述一个系统, 我们可以对运行做限制以排除一些部分这样的无穷状态序列。

定义 4.3 一个Büchi自动机是一个五元组 $(\Sigma, S, \Delta, I, F)$ 其中 (Σ, S, Δ, I) 是一个标号迁移系统, $F \subseteq S$ 为状态集。

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

$$\text{inf}(\pi) \cap F \neq \emptyset$$

可接受字符串与自动机语言： 一个无穷字符串 $w = [a_i]_{i \geq 1}$ 是可接受的当且仅当存在 w 上的可接受运行。 \mathcal{B} 的可接受字符串的集合为 \mathcal{B} 的语言，记作 $\mathcal{L}(\mathcal{B})$ 。

语言非空问题： 自动机 \mathcal{B} 的语言为空，即 $\mathcal{L}(\mathcal{B}) = \emptyset$ ，记作 $\mathcal{B} \equiv \emptyset$ 。

命题 4.2 给定 Büchi 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 。令 $R = \{(s, s') \mid (s, a, s') \in \Delta\}$ 。 $\mathcal{B} \equiv \emptyset$ 当且仅当公平 Kripke 模型 $K = \langle S, R, I, \{F\} \rangle$ 为空当且仅当 $\text{EmpChecking}(K)$ 为 true 当且仅当 $\text{EmpCheckingDDFS}(K)$ 为 true。

自动机的运算： 自动机的运算包括交、并和补。 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 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 的补自动机，记作 $A' = \neg A$ ，当前仅当 $\mathcal{L}(A') = \Sigma^\omega \setminus \mathcal{L}(A)$ 。

定义 4.4 Büchi 自动机是交封闭的，当且仅当对任意 Büchi 自动机 B_1 和 B_2 ，存在一个 Büchi 自动机 B 使得 $\mathcal{L}(B) = \mathcal{L}(B_1) \cap \mathcal{L}(B_2)$ 。

命题 4.3 Büchi 自动机是交封闭的。

以下是一种构造交自动机的算法。

给定两个 Büchi 自动机 $B_1 = \langle \Sigma, S_1, \Delta_1, I_1, F_1 \rangle$, $B_2 = \langle \Sigma, S_2, \Delta_2, I_2, F_2 \rangle$ 。

定义 $S = S_1 \times S_2 \times \{0, 1, 2\}$ 且 Δ 为以下集合的并集。

$$\begin{aligned} & \overline{\{(s_1, s_2, i), a, ((s'_1, s'_2, i)) \mid (s_1, a, s'_1) \in \Delta_1, (s_2, a, s'_2) \in \Delta_2, i \in \{0, 1\}\}} \\ & \overline{\{(s_1, s_2, i), a, ((s'_1, s'_2, i + 1)) \mid (s_1, a, s'_1) \in \Delta_1, (s_2, a, s'_2) \in \Delta_2, i \in \{0, 1\}, s_{i+1} \in F_{i+1}\}} \\ & \overline{\{(s_1, s_2, 2), a, ((s'_1, s'_2, 0)) \mid (s_1, a, s'_1) \in \Delta_1, (s_2, a, s'_2) \in \Delta_2\}} \end{aligned}$$

定义 $B_1 \cap B_2 = \langle \Sigma, S, \Delta, I_1 \times I_2 \times \{0\}, S_1 \times S_2 \times \{2\} \rangle$ 。则 $\mathcal{L}(B_1 \cap B_2) = \mathcal{L}(B_1) \cap \mathcal{L}(B_2)$ 。

定义 4.5 Büchi 自动机是并封闭的，当且仅当对任意 Büchi 自动机 B_1 和 B_2 ，存在一个 Büchi 自动机 B 使得 $\mathcal{L}(B) = \mathcal{L}(B_1) \cup \mathcal{L}(B_2)$ 。

命题 4.4 Büchi 自动机是并封闭的。

以下是一种构造并自动机的算法。

给定两个 Büchi 自动机 $B_1 = \langle \Sigma, S_1, \Delta_1, I_1, F_1 \rangle$, $B_2 = \langle \Sigma, S_2, \Delta_2, I_2, F_2 \rangle$ 且 $S_1 \cap S_2 = \emptyset$ 。

定义 $B_1 \cup B_2 = \langle \Sigma, S_1 \cup S_2, \Delta_1 \cup \Delta_2, I_1 \cup I_2, F_1 \cup F_2 \rangle$ 。则 $\mathcal{L}(B_1 \cup B_2) = \mathcal{L}(B_1) \cup \mathcal{L}(B_2)$ 。

定义 4.6 Büchi 自动机是补封闭的，当且仅当对任意 Büchi 自动机 B ，存在一个 Büchi 自动机 B' 使得 $\mathcal{L}(B') = \Sigma^\omega \setminus \mathcal{L}(B)$ 。

命题 4.5 Büchi 自动机是补封闭的。

以下是一种构造补自动机的算法。

给定Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。我们将 $\alpha \in \Sigma^*$ 根据 $p, q \in S$ ，按以下条件进行分类：可连接 p, q 且可经过一个 F 状态的；可连接 p, q 且不能经过 F 状态的；不能连接 p, q 的。记 n 为状态的数目。类型的总数不超过 3^{n^2} 。记这些类型为 t_1, \dots, t_m 。

由Ramsey定理（见第一章）可得，任意无穷串可写为 $t_i(t_j)^\omega$ 的形式。

因而 $\Sigma^\omega = \bigcup \{t_i(t_j)^\omega \mid 1 \leq i, j \leq m\}$ 。

由分类的构造知， $t_i(t_j)^\omega \subseteq \mathcal{L}(B)$ 或 $t_i(t_j)^\omega \subseteq \Sigma^\omega \setminus \mathcal{L}(B)$ 。

因而 $\Sigma^\omega \setminus \mathcal{L}(B) = \bigcup \{t_i(t_j)^\omega \mid 1 \leq i, j \leq m, t_i(t_j)^\omega \not\subseteq \mathcal{L}(B)\}$ 。

由每类 t_i 是一个有穷字上的自动机的语言知， $t_i(t_j)^\omega$ 是一个Büchi 自动机的语言。

定义 $A_{i,j}$ 使得其语言为 $t_i(t_j)^\omega$ 。

定义 $B' = \bigcup \{A_{i,j} \mid 1 \leq i, j \leq m, A_{i,j} \cap B \equiv \emptyset\}$ 。则 B' 是Büchi 自动机 B 的补自动机。

语言包含问题： 自动机 A 的语言是否包含于自动机 B 的语言，即 $\mathcal{L}(A) \subseteq \mathcal{L}(B)$ 是否成立，可通过自动机的交和补运算转换成语言非空问题。 $\mathcal{L}(A) \subseteq \mathcal{L}(B)$ 当且仅当 $\mathcal{L}(A) \cap (\Sigma^\omega \setminus \mathcal{L}(B)) = \emptyset$ 当且仅当 $A \cap \neg B \equiv \emptyset$ 。

有穷字符串自动机与Büchi 自动机的关系： 有穷字符串自动机的形式与Büchi 自动机相同。设 $\mathcal{A} = \langle \Sigma, S, \Delta, I, F \rangle$ 为有穷字符串自动机。一个字符串 $w = w_1 \cdots w_n \in \Sigma^*$ 是 \mathcal{A} 接受的当且仅当存在 $r_0 \in I, r_n \in F, r_1, \dots, r_{n-1} \in S$ 且对所有 $0 \leq i \leq n-1, (r_i, w_{i+1}, r_{i+1}) \in \Delta$ 。 \mathcal{A} 的可接受字符串的集合为 \mathcal{A} 的语言。

§4.2.1 自动机的可接受运行条件的设置

我们可以根据不同的需要设置不同的可接受运行的条件。这些条件的主要类型有以下几种。

泛Büchi 自动机：

定义 4.7 一个泛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， $F \subseteq 2^S$ 为状态集的集合。

泛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 $\langle \Sigma, S, \Delta, I \rangle$ 的运行。 \mathcal{B} 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的当且仅当 $\forall f \in F. (inf(\pi) \cap f \neq \emptyset)$ 。

命题 4.6 给定Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。定义泛Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, \{F\} \rangle$ 。则 $L(B') = L(B)$ 。

Streett 自动机：

定义 4.8 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， $F \subseteq 2^S \times 2^S$ 为状态集合对的集合。

Streett 自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 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 $(f, g) \in F, inf(\pi) \cap f \neq \emptyset \rightarrow inf(\pi) \cap g \neq \emptyset$ 。

命题 4.7 给定泛Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。定义Streett自动机 $B' = \langle \Sigma, S, \Delta, I, F' \rangle$ 其中 $F' = \{(S, f) \mid f \in F\}$ 。则 $L(B') = L(B)$ 。

Parity自动机:

定义 4.9 一个Parity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, $F : S \rightarrow N$ (其中 N 为自然数集合) 为优先级函数。

Parity自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 的运行。 \mathcal{B} 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的当且仅当以下集合的最小元为偶数: $\{F(s) \mid s \in \text{inf}(\pi)\}$ 。

命题 4.8 给定Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。定义Parity自动机 $B' = \langle \Sigma, S, \Delta, I, F' \rangle$ 其中 F' 定义如下: 若 $s \in F$ 则 $f(s) = 0$ 否则 $f(s) = 1$ 。则 $L(B') = L(B)$ 。

Rabin自动机:

定义 4.10 一个Rabin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, $F \subseteq 2^S \times 2^S$ 为状态集合对的集合。

Rabin自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, F \rangle$ 上的运行就是标号迁移系统 $\langle \Sigma, S, \Delta, I \rangle$ 的运行。 \mathcal{B} 上的运行 $\pi = [s_i]_{i \geq 0}$ 是可接受的当且仅当存在 $(f, g) \in F$, $\text{inf}(\pi) \cap f \neq \emptyset \wedge \text{inf}(\pi) \cap g = \emptyset$ 。

命题 4.9 给定Parity自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。设 $F(s)$ 的最大值为 $2n$ 。记 $\{s \mid f(s) = k\}$ 为 f_k 。定义Rabin自动机 $B' = \langle \Sigma, S, \Delta, I, F' \rangle$ 其中 $F' = \{(f_{2i}, \cup_{j=0}^{i-1} f_{2j+1}) \mid i = 0, \dots, n\}$ 。则 $L(B') = L(B)$ 。

Muller 自动机:

定义 4.11 一个Muller自动机是一个五元组 $\langle \Sigma, S, \Delta, I, F \rangle$ 其中 $\langle \Sigma, S, \Delta, I \rangle$ 是一个标号迁移系统, $F \subseteq 2^S$ 为状态集的集合。

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

命题 4.10 给定Streett自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。定义Muller自动机 $B' = \langle \Sigma, S, \Delta, I, F' \rangle$ 其中 $F' = \cap_{(f,g) \in F} (\{Y \mid Y \cap f = \emptyset \vee Y \cap g \neq \emptyset\})$ 。则 $L(B') = L(B)$ 。

命题 4.11 给定Rabin自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。定义Muller自动机 $B' = \langle \Sigma, S, \Delta, I, F' \rangle$ 其中 $F' = \cup_{(f,g) \in F} (\{Y \mid Y \cap f \neq \emptyset, Y \cap g = \emptyset\})$ 。则 $L(B') = L(B)$ 。

命题 4.12 给定Muller自动机 $\mathcal{B} = \langle \Sigma, S, \Delta, I, \{F_0, \dots, F_n\} \rangle$ 。定义 $S' = S \cup \cup_{i=0}^n \{i\} \times F_i \times 2^{F_i}$ 且 Δ' 为 Δ 与以下集合的并集。

$$\begin{aligned} & \{(s, a, (i, s', \emptyset)) \mid (s, a, s') \in \Delta, s' \in F_i, 0 \leq i \leq n\} \\ & \{((i, s, R), a, (i, s', R \cup \{s\})) \mid (s, a, s') \in \Delta, R \neq F_i, s, s' \in F_i, 0 \leq i \leq n\} \\ & \{((i, s, F_i), a, (i, s', \emptyset)) \mid (s, a, s') \in \Delta, s, s' \in F_i, 0 \leq i \leq n\} \end{aligned}$$

定义Büchi自动机 $B' = \langle \Sigma, S', \Delta', I, \cup_{i=0}^n \{i\} \times F_i \times \{F_i\} \rangle$ 。则 $L(B') = L(B)$ 。

自动机表达能力的等价： 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。

命题 4.13 Büchi 自动机、泛Büchi 自动机、Muller 自动机、Streett 自动机、Rabin 自动机和Parity 自动机的表达能力是相同的。

§4.2.2 确定型自动机与非确定型自动机

自动机称为确定型自动机当且仅当其标号迁移系统是确定型。不受确定型条件限制的自动机亦称为非确定型自动机。

命题 4.14 确定型Büchi 自动机不是补封闭的。

命题 4.15 确定型Muller 自动机、确定型Streett 自动机、确定型Rabin 自动机和确定型Parity 自动机是补封闭的且其表达能力与非确定型Muller 自动机相同。

§4.2.3 自动机与Kripke模型

定义 4.12 给定泛Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 和公平标号Kripke模型 K 。 B 与 K 是等价的，记作 $B \equiv K$ ，当且仅当 $\mathcal{L}(B) = \mathcal{L}(K)$ ，或对任意 $[w_i]_{i \geq 0} \in \mathcal{L}(B)$ 存在 $[w'_i]_{i \geq 0} \in \mathcal{L}(K)$ 使得对所有 $i \geq 0$ 都有 $\{w_i\} = w'_i \cap \Sigma$ 且对任意 $[w'_i]_{i \geq 0} \in \mathcal{L}(K)$ 存在 $[w_i]_{i \geq 0} \in \mathcal{L}(B)$ 使得对所有 $i \geq 0$ 都有 $\{w_i\} = w'_i \cap \Sigma$ 。

对于自动机我们可以构造与之等价的公平标号Kripke模型。反之亦然。

公平标号Kripke模型到泛Büchi 自动机的构造： 给定 AP 上的公平标号Kripke模型 $K = \langle S, R, I, L, F \rangle$ 。我们构造与之等价的泛Büchi 自动机如下。

定义 $\Sigma = 2^{AP}$ 。

定义 $\Delta = \{(s, a, s') \mid (s, s') \in R, a = L(s)\}$ 。

定义泛Büchi 自动机 $B_M = \langle \Sigma, S, \Delta, I, \{[\varphi] \mid \varphi \in F\} \rangle$ 。则 $\mathcal{L}(B) = \mathcal{L}(K)$ 。

标号Kripke模型到泛Büchi 自动机的构造： 标号Kripke模型 $K = \langle S, R, I, L \rangle$ 可以看成是特殊的公平标号Kripke模型 $K' = \langle S, R, I, L, \{\} \rangle$ 。可以用以上方法构造与之等价的泛Büchi 自动机。

泛Büchi 自动机到公平标号Kripke模型的构造： 给定泛Büchi 自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 且 $F = \{f_1, \dots, f_n\}$ 。我们构造与之等价的公平标号Kripke模型如下。

定义 $AP = \Sigma \cup \{p_1, \dots, p_n\}$ 。

定义 $S' = \Delta$ 。

定义 $R' = \{((s, a, s'), (s', b, s'')) \mid (s, a, s'), (s', b, s'') \in S'\}$ 。

定义 $I' = \{(s, a, s') \mid (s, a, s') \in S', s \in I\}$

定义 $L((s, a, s')) = \{a\} \cup \{p_i \mid s \in f_i, i = 1, \dots, n\}$ 。

定义 $F' = \{p_1, \dots, p_n\}$ 。

定义 AP 上的公平标号Kripke模型 $K = \langle S', R', I', L, F' \rangle$ 。

定义 $\zeta(X)$ 如下：若存在唯一的 a 使得 $X \cap \Sigma = \{a\}$ 则 $\zeta(X) = a$ ，否则 $\zeta(X) = \emptyset$ 。

定义 $\zeta([w_i]_{i \geq 0}) = [\zeta(w_i)]_{i \geq 0}$ 。
 定义 $\zeta(W) = \{\zeta(w) \mid w \in W\}$ 。
 则 $\zeta(L(K)) = L(B)$ 。

命题 4.16 给定泛Büchi自动机 $B = \langle \Sigma, S, \Delta, I, F \rangle$ 。令 $R = \{(s, s') \mid (s, a, s') \in \Delta\}$ 。 $B \equiv \emptyset$ 当且仅当公平Kripke模型 $K = \langle S, R, I, F \rangle$ 为空当且仅当 $\text{EmpChecking}(K)$ 为 true。

§4.3 时间迁移系统与时间自动机

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长短或两个状态之间的时间变化。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。

时钟公式： 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(有理数)的集合。时钟公式的集合 $\Phi(X)$ 由以下语法给出。

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

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 $R_{\geq 0}$ (非负实数) 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 $[Y \rightarrow t]v$ 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' 。

定义 4.13 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 为有穷状态集合， 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， $I \in S$ 为模型初始状态的集合。

给定时间迁移系统 $M = \langle \Sigma, S, X, \Delta, I \rangle$ 。

状态空间： 模型状态由两部分组成，一部分是状态 $s \in S$ ，另一部分为时间变量的状态，称为时钟赋值。记 $V = X \rightarrow R$ 为时钟赋值的集合。 M 的状态空间为 $S \times V$ 。

时间迁移与动作迁移： 设 t 为时间常量。定义 $(s, v) \xrightarrow{t} (s, v')$ 当且仅当 $v' = v + t$ 。设 $\sigma \in \Sigma$ 。定义 $(s, v) \xrightarrow{\sigma} (s', v')$ 当且仅当存在 $(s, \sigma, \lambda, \varphi, s') \in \Delta$ 使得 $v \models \varphi$ 且 $v' = [\lambda \rightarrow 0]v$ 。

迁移关系： 定义 $(s, v) \xrightarrow{(\sigma, t)} (s', v')$ 当且仅当存在 v'' 使得 $(s, v) \xrightarrow{t} (s, v'')$ 且 $(s, v'') \xrightarrow{\sigma} (s', v')$ 。 M 的迁移关系 \rightarrow 是 $(S \times V)^2$ 的子集。 $(s, v) \rightarrow (s', v')$ 当且仅当存在 (σ, t) 使得 $(s, v) \xrightarrow{(\sigma, t)} (s', v')$ 。

时间字符串上： 一个时间字符串为 $\Sigma \times R$ 上的无穷序列。为方便起见，时间字符串也表示为 $(\zeta, \tau) = ([\zeta_i]_{i \geq 1}, [\tau_i]_{i \geq 1}) \in \Sigma^\omega \times R^\omega$ 其中 τ 满足对所有 i , $\tau_{i+1} > \tau_i$ 且对任意 $t \in R$ 存在 i , $\tau_i > t$ 。为方便建模，有些模型将 $\tau_{i+1} > \tau_i$ 的要求弱化为 $\tau_{i+1} \geq \tau_i$ 。

时间字符串上的运行： 设 $(\zeta, \tau) = [(\zeta_i, \tau_i)]_{i \geq 1} \in (\Sigma \times R)^\omega$ 为时间字符串。定义 $\tau_0 = 0$ 。时间迁移系统 $\langle \Sigma, S, X, \Delta, I \rangle$ 在 (ζ, τ) 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

$$\begin{array}{l} \hline s_0 \in I, \\ v_0 = [X \rightarrow 0]v_0, \\ \text{对所有 } i \geq 0, (s_i, v_i) \xrightarrow{(\zeta_{i+1}, \tau_{i+1} - \tau_i)} (s_{i+1}, v_{i+1}). \end{array}$$

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

可达状态： 给定时间迁移系统 $M = \langle \Sigma, S, X, \Delta, I \rangle$ 。一个状态 (s, v) 可达当且仅当存在一个运行 r 和 r 中的一步迁移 $(s, v') \xrightarrow{(\sigma, t)} (s', v'')$ 以及 $t' \geq 0$ 使得 $(s, v') \xrightarrow{t'} (s, v)$ 且 $(s, v) \xrightarrow{(\sigma, t-t')} (s', v'')$ 。这里不考虑可达之后必然进入不存在满足迁移条件的一类状态的情况。

可达性问题和安全性问题： 可达性问题是指给定一个模型和一个状态集合，判断模型是否有可达状态满足给定的状态集合。安全性问题是指给定一个模型和一个状态集合，判断模型的所有可达状态是否都满足给定的状态集合。尽管时间迁移系统的可达状态集合通常是无穷的，时间迁移系统的状态可达性问题是可判定的。

定义 4.14 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 \mathcal{B} 的运行 $r = (s, v)$ 是可接受的，当且仅当 $\text{inf}(r) \cap F \neq \emptyset$ ，其中 $\text{inf}(r)$ 代表无限多次出现在运行 r 中的状态的集合。

可接受时间字符串与自动机语言： 一个时间字符串 (σ, τ) 是可接受的当且仅当存在 (σ, τ) 的可接受运行。 \mathcal{B} 的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。

§4.4 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 $X = \{x_1, \dots, x_n\}$ 为实数变量的集合。每个实数变量都是时间的函数。给定 $x \in X$ 。记 \dot{x} 为 x 的导函数。记 $\dot{X} = \{\dot{x}_1, \dots, \dot{x}_n\}$ 。记 $\Phi(X)$ 为 X 上谓词的集合。记 $\Theta(X) : X \hookrightarrow (R^n \rightarrow R)$ 为偏函数的集合。

定义 4.15 一个混成迁移系统(也称为混成自动机)是一个六元组 $\langle \Sigma, S, X, \Delta, I, \text{flow} \rangle$ 其中 Σ 为字母表， S 为有穷状态集合， X 为实数变量集合， $\Delta \subseteq S \times \Sigma \times \Theta(X) \times \Phi(X) \times S$ 为迁移关系， $I \in S \times (2^R)^n$ 为模型初始状态的集合， $\text{flow} : S \rightarrow \Phi(X \cup \dot{X})$ 为每个状态标上一个变量变化的约束条件。

给定混成迁移系统 $\langle \Sigma, S, X, \Delta, I, \text{flow} \rangle$ 。

状态空间： 模型状态由两部分组成，一部分是状态 $s \in S$ ，另一部分为变量的状态，称为赋值。记 $V = X \rightarrow R$ 为赋值的集合。 M 的状态空间为 $S \times V$ 。

时间迁移与动作迁移： 设 $\delta \in R_{\geq 0}$ 为时间常量。定义 $(s, v) \xrightarrow{\delta} (s, v')$ 当且仅当存在一个可导函数 $f : [0, \delta] \rightarrow R^n$ 和其一阶导数 $\dot{f} : (0, \delta) \rightarrow R^n$ 满足 $f(0) = v, f(\delta) = v'$ 且

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

设 $\sigma \in \Sigma$ 。定义 $(s, v) \xrightarrow{\sigma} (s', v')$ 当且仅当存在 $(s, \sigma, \lambda, \varphi, s') \in \Delta$ 且 $\{z_1, \dots, z_k\} = dom(\lambda)$ 使得 $v \models \varphi$ 且 $v' = v[z_1/\lambda(z_1)(v)] \cdots [z_k/\lambda(z_k)(v)]$ 。

迁移关系： 定义 $(s, v) \xrightarrow{(\sigma, t)} (s', v')$ 当且仅当存在 v'' 使得 $(s, v) \xrightarrow{t} (s, v'')$ 且 $(s, v'') \xrightarrow{\sigma} (s', v')$ 。 M 的迁移关系 \rightarrow 是 $(S \times V)^2$ 的子集。 $(s, v) \rightarrow (s', v')$ 当且仅当存在 (σ, t) 使得 $(s, v) \xrightarrow{(\sigma, t)} (s', v')$ 。

时间字符串上的运行： 设 $(\zeta, \tau) = [(\zeta_i, \tau_i)]_{i \geq 1} \in (\Sigma \times R)^\omega$ 为时间字符串。定义 $\tau_0 = 0$ 。时间迁移系统 $\langle \Sigma, S, X, \Delta, I \rangle$ 在 (ζ, τ) 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

$$\begin{array}{l} (s_0, v_0) \in I, \\ \text{对所有 } i \geq 0, (s_i, v_i) \xrightarrow{(\zeta_{i+1}, \tau_{i+1} - \tau_i)} (s_{i+1}, v_{i+1}). \end{array}$$

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 (ζ, τ) 使得 r 是混成迁移系统在 (ζ, τ) 上的一个运行。

可达状态： 给定混成迁移系统 $M = \langle \Sigma, S, X, \Delta, I, flow \rangle$ 。一个状态 (s, v) 可达当且仅当存在一个运行 r 和 r 中的一步迁移 $(s, v') \xrightarrow{(\sigma, t')} (s', v'')$ 以及 $t' \geq 0$ 使得 $(s, v') \xrightarrow{t'} (s, v)$ 且 $(s, v) \xrightarrow{(\sigma, t')} (s', v'')$ 。

可达性问题和安全性问题： 可达性问题是指给定一个模型和一个状态集合，判断模型是否有可达状态满足给定的状态集合。安全性问题是指给定一个模型和一个状态集合，判断模型的所有可达状态是否都满足给定的状态集合。

§4.5 Petri网模型

Petri网模型的基本要素有库所、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。

定义 4.16 一个Petri网模型是一个四元组 $\langle P, T, F, m_0 \rangle$ 其中 P 为库所的有穷集合， T 为迁移的有穷集合， $F \subseteq (P \times T) \cup (T \times P)$ 为边的集合， $m_0 : P \rightarrow N$ 为库所集合到自然数集合的函数，表示初始状态。

给定 $M = \langle P, T, F, m_0 \rangle$ 。

对 $t \in T$ ，定义 ${}^\circ p(t) = \{p \in P \mid (p, t) \in F\}$ 和 $p^\circ(t) = \{p \in P \mid (t, p) \in F\}$ 。

${}^\circ p(t)$ 和 $p^\circ(t)$ 分别为 t 的输入库所和输出库所的集合。由 ${}^\circ p()$ 和 $p^\circ()$ 可重构 F 。

状态空间： 模型状态是所有库所的状态的组合，表示为对库所进行赋值的一个函数。模型的状态空间为 $V = P \rightarrow N$ 。

可执行迁移： 迁移 $t \in T$ 在状态 m 是可执行的，当且仅当 $\forall p \in {}^\circ p(t). (m(p) \geq 1)$ 。

死锁状态： 若在状态 m 没有可执行迁移，则称 m 为死锁状态。

迁移关系(交错语义)： 定义 $\alpha_0(p, t) = 1$ 当且仅当 $p \in {}^\circ t$ 。定义 $\alpha_1(p, t) = 1$ 当且仅当 $p \in p^\circ(t)$ 。定义 $m \xrightarrow{t} m'$ 当且仅当迁移 $t \in T$ 在状态 m 可执行且

$$\forall p \in P. (m'(p) = m(p) - \alpha_0(p, t) + \alpha_1(p, t)).$$

M 的迁移关系 \rightarrow 是 V^2 的子集。 $m \rightarrow m'$ 当且仅当存在可执行迁移 t 使得 $m \xrightarrow{t} m'$ 或不存在可执行迁移且 $m = m'$ 。

运行： 状态空间 V 上的一个无穷序列 $[m_i]_{i \geq 0}$ 是 M 的一个运行，当且仅当对所有 i 有 $m_i \rightarrow m_{i+1}$ 。

可达状态： 给定 $M = \langle P, T, F, m_0 \rangle$ 。一个状态 m 是可达的当且仅当 $m_0 \xrightarrow{*} m$ 。 M 的可达状态集合为 $\{m \mid m_0 \xrightarrow{*} m\}$ 。

可达性问题： 状态的可达性问题是指给定一个模型和一个状态，判断这个状态是否可达。尽管Petri网模型的可达状态集合通常是无穷的，Petri网模型的状态可达性问题是可判定的。

活性性质： 迁移的活性是指迁移是否可执行或者以至于是否能执行无限多次。根据执行次数的区别可定义 L_1 活性(迁移有可能执行)和 L_2 活性(迁移有可能执行任意多次)等性质。

有穷状态模型： Petri网模型 $\langle P, T, W, m_0 \rangle$ 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。

加权Petri网模型： 为方便建模，我们可以对Petri网模型的边增加权重，建立加权Petri网模型。

定义 4.17 一个加权Petri网模型是一个四元组 $\langle P, T, W, m_0 \rangle$ 其中 P, T 和 m_0 同于以上Petri网模型， $W : (P \times T) \cup (T \times P) \rightarrow N$ 为具有权重的边的集合。

给定 $M = \langle P, T, W, m_0 \rangle$ 。 M 的状态空间同于以上Petri网模型。迁移 $t \in T$ 在状态 m 是可执行的，当且仅当 $\forall p \in {}^\circ p(t). (m(p) \geq W(p, t))$ 。 $m \xrightarrow{t} m'$ 当且仅当迁移 $t \in T$ 在状态 m 可执行且

$$\forall p \in P. (m'(p) = m(p) - W(p, t) + W(t, p)).$$

M 的迁移关系 \rightarrow ，运行集合和可达状态集合的定义类似于普通Petri网模型。加权Petri网模型并不增加表达能力。对加权Petri网模型我们可以构造等效的(可以映射到同样状态序列的)普通Petri网模型。

命题 4.17 每一个加权Petri网模型都可以转换成一个等效的普通Petri网模型。

容量限制的Petri网模型： 我们可以对Petri网模型的库所进行容量限制，建立容量限制的Petri网模型。

定义 4.18 一个容量限制的Petri网模型是一个五元组 $\langle P, T, F, K, m_0 \rangle$ 其中 $\langle P, T, F, m_0 \rangle$ 为普通Petri网模型， $K: P \rightarrow N$ 为容量限制函数。

给定 $M = \langle P, T, F, K, m_0 \rangle$ 。M的状态空间同于以上Petri网模型。迁移 $t \in T$ 在状态 m 是可执行的，当且仅当 $\forall p \in p^\circ(t).(m(p) \geq 1)$ 且 $\forall p \in p^\circ(t).(m(p) - \alpha_0(p, t) + \alpha_1(p, t) \leq K(p))$ 。 $m \xrightarrow{t} m'$ 当且仅当迁移 $t \in T$ 在状态 m 可执行且

$$\forall p \in P.(m'(p) = m(p) - \alpha_0(p, t) + \alpha_1(p, t)).$$

M的迁移关系 \rightarrow ，运行集合和可达状态集合的定义类似于普通Petri网模型。

命题 4.18 每一个容量限制的Petri网模型都可以转换成一个等效的普通Petri网模型。

§4.6 通信系统

通信系统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。

§4.6.1 通道

定义 4.19 给定一个类型 $\langle \Lambda, n \rangle$ 其中 Λ 为字母表， n 为自然数，表示通道容量。一个类型为 $\langle \Lambda, n \rangle$ 的通道 m ，记作 $m \in \langle \Lambda, n \rangle$ ，是一个值为 $\bigcup_{i=0}^n \{ \langle x_1, \dots, x_i \mid x_i \in \Lambda \rangle \}$ 的变量。

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

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

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

$$\begin{array}{l} \hline a = \epsilon \\ a = m!s \text{ 且 } |\sigma(m)| < n \\ a = m?s \text{ 且 } |\sigma(m)| > 0 \text{ 且 } s = HEAD(\sigma(m)) \\ \hline \end{array}$$

给定 $\sigma \in \Sigma, m \in \langle \Lambda, n \rangle, a \in \alpha(m)$ 。用 $\sigma \xrightarrow{a} \sigma'$ 表示 a 在 σ 可执行且以下成立。

$$\begin{array}{l} \hline \text{若 } a = \epsilon, \quad \text{则 } \sigma' = \sigma \\ \text{若 } a = m!s, \quad \text{则 } \sigma' = \sigma[m/\sigma(m) \vdash s] \\ \text{若 } a = m?s, \quad \text{则 } \sigma' = \sigma[m/TAIL(\sigma(m))] \\ \hline \end{array}$$

§4.6.2 通信单元

定义 4.20 一个通信单元是一个四元组 $\langle Q, C, \Delta, q_0 \rangle$ 其中 Q 为状态的有穷集合, C 为通道的有穷集合, $\Delta \subseteq Q \times (\alpha(C) \cup \{\epsilon\}) \times Q$ 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。

迁移关系: 迁移 (q, a, q') 在 (s, σ) 可执行, 当且仅当 $q = s$ 且 a 在 σ 可执行。系统状态的迁移关系 \rightarrow 为 $(Q \times \Sigma)^2$ 的子集。 $(q, \sigma) \rightarrow (q', \sigma')$ 当且仅当存在在状态 (q, σ) 的可执行迁移 (q, a, q') 使得 $\sigma \xrightarrow{a} \sigma'$ 或不存在可执行迁移且 $(q', \sigma') = (q, \sigma)$ 。

运行: 一个运行是 $Q \times \Sigma$ 上的一个无穷序列 $[(s_i, \sigma_i)]_{i \geq 0}$ 满足 $s_0 = q_0, \forall m \in C. (\sigma_0(m) = \langle \rangle)$ 且对任意 $i \geq 0, (s_i, \sigma_i) \rightarrow (s_{i+1}, \sigma_{i+1})$ 。通信单元 P 的运行集合记为 $[[P]]$ 。

§4.6.3 通信系统

为方便建模, 通常用一组通信单元构建通信系统模型。

定义 4.21 一个通信系统 $\{P_1, \dots, P_n\}$ 是由通信单元 $P_1 = \langle Q_1, C_1, \Delta_1, q_{10} \rangle, \dots, P_n = \langle Q_n, C_n, \Delta_n, q_{n0} \rangle$ 组成的集合, 其中 Q_1, \dots, Q_n 为两两不相交的集合。

系统状态: 定义 $Q_1 \otimes \dots \otimes Q_n = \{q_1, \dots, q_n \mid q_1 \in Q_1, \dots, q_n \in Q_n\}$ 。通信系统的系统状态的集合为 $(Q_1 \otimes \dots \otimes Q_n) \times \Sigma$ 。

可执行迁移: 一个迁移 $(q, a, q') \in \Delta_1 \cup \dots \cup \Delta_n$ 在状态 (Q, σ) 可执行当且仅当 $q \in Q$ 且 a 在 σ 可执行。

迁移关系: 给定 $a \in \alpha(C) \cup \{\epsilon\}$ 。 $(z, \sigma) \xrightarrow{a} (z', \sigma')$ 当且仅当存在在状态 (z, σ) 的可执行迁移 $(q, a, q') \in \Delta_1 \cup \dots \cup \Delta_n$ 使得 $z' = (z \setminus \{q\}) \cup \{q'\}$ 且 $\sigma \xrightarrow{a} \sigma'$ 。系统状态的迁移关系 \rightarrow 为 $((Q_1 \otimes \dots \otimes Q_n) \times \Sigma)^2$ 的子集。 $(z, \sigma) \rightarrow (z', \sigma')$ 当且仅当存在在状态 (z, σ) 的可执行迁移 (q, a, q') 使得 $(z, \sigma) \xrightarrow{a} (z', \sigma')$ 或不存在可执行迁移且 $(z', \sigma') = (z, \sigma)$ 。

运行: 设 $C_1 \cup \dots \cup C_n = \{m_1, \dots, m_k\}$ 。一个运行是 $(Q_1 \otimes \dots \otimes Q_n) \times \Sigma$ 上的一个无穷序列 $[(z_i, \sigma_i)]_{i \geq 0}$ 满足 $z_0 = \{q_0, \dots, q_n\}, \sigma_0(m_1) = \dots = \sigma_0(m_k) = \langle \rangle$ 且对任意 $i \geq 0, (z_i, \sigma_i) \rightarrow (z_{i+1}, \sigma_{i+1})$ 。通信系统 M 的运行集合记为 $[[M]]$ 。

通信系统与通信单元 由多个通信单元构建的通信系统并不比通信单元具有更强的表达能力。给定通信系统, 我们可以构造行为等价的通信单元。

定义 $g(a)$ 如下。

若 a 不是一个集合, 则 $g(a) = \{a\}$ 。若 $a = \{a_1, \dots, a_j\}$, 则 $g(a) = g(a_1) \cup \dots \cup g(a_j)$ 。

定义 $g(a, b) = g(a) \cup g(b)$ 。

定义 $\|a\|$ 如下。

若 a 不是一个集合, 则 $\|a\| = 1$ 。若 $a = \{a_1, \dots, a_j\}$, 则 $\|a\| = \|a_1\| + \dots + \|a_j\|$ 。

定义 $|x|$ 为集合 x 的元素个数。

给定一个集合 Q 。若 $\forall q \in Q. (\|g(q)\| = \|q\|)$, 则称 Q 为不可压缩集合。

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 和 Q_2 不相交且 $Q_1 \otimes Q_2$ 为不可压缩集合。

定义 $Q = \{g(q_1, q_2) \mid q_1 \in Q_1, q_2 \in Q_2\}$ 。

定义 $\Delta = \{(g(q_1, q_2), a, g(q'_1, q_2)) \mid (q_1, a, q'_1) \in \Delta_1\} \cup \{(g(q_1, q_2), a, g(q_1, q'_2)) \mid (q_2, a, q'_2) \in \Delta_2\}$ 。

定义 P_1 和 P_2 的并发系统如下。

$$P_1 \parallel P_2 = \langle Q, C_1 \cup C_2, \Delta, g(q_{10}, q_{20}) \rangle.$$

由以上定义容易看出并发算子满足结合率和交换率。

命题 4.19 给定通信系统 $M = \{P_1, \dots, P_n\}$ 且其中 Q_1, \dots, Q_n 为两两不相交且两两的 \otimes 操作结果为不可压缩集合。定义通信单元 $P = P_1 \parallel \dots \parallel P_n$ 。则 $[[P]] = [[M]]$ 。

§4.6.4 通信单元与卫式迁移模型的等价

考虑一种关于通信单元运行中 Q 的序列的等价关系。

给定通信单元 $P = \langle Q, C, \Delta, q_0 \rangle$ 。

给定 $B = (F, P)$ 和 V 且 $Q \subseteq F$ 和 $pc \in V$ 。

给定 (B, V) 上的卫式迁移模型 $M = (T, \Theta)$ 和 B 的解释 $I = (D, I_0)$ 。

给定 $\zeta: Q \rightarrow D$ 为单射函数。

定义 4.22 P 与 M 是 ζ 计算等价的, 当且仅当对任意 P 的计算 $[(s_i, \sigma_i)]_{i \geq 0}$ 存在 M 的计算 $[\sigma'_i]_{i \geq 0}$ 满足 $\forall i \geq 0. (\sigma'_i \models_I pc = s_i)$ 且对任意 M 的计算 $[\sigma'_i]_{i \geq 0}$ 存在 P 的计算 $[(\zeta^{-1}(\sigma'(pc)), \sigma_i)]_{i \geq 0}$ 。

通信单元的卫式迁移模型的构造: 给定通信单元 $P = \langle Q, C, \Delta, q_0 \rangle$ 。设 $C = \{m_1, \dots, m_k\}$ 且 m_i 的类型为 $\langle \Lambda_i, n_i \rangle$ 。

定义 $B = (F, P)$ 其中 $F = \Lambda_1 \cup \dots \cup \Lambda_k \cup Q \cup \{0, 1, +, -\}$ 且 $P = \{>, =, <\}$ 。

定义 $V = \{v_{i,j} \mid i \in \{1, \dots, k\}, j \in \{0, \dots, n_i - 1\}\} \cup \{pc\}$ 。

我们构造 (B, V) 上的卫式迁移模型 (T, Θ) 如下。

对每个 (q, a, q') 构造如下迁移并将其加入迁移集合 T 。

(1) 若 $a = \epsilon$, 则构造迁移

$$pc = q \longrightarrow (pc) := (q').$$

(2) 若 $a = m_i!s$, 则构造迁移

$$pc = q \wedge len_i < n_i \longrightarrow (v_{i,len_i}, len_i, pc) := (s, len_i + 1, q').$$

(3) 若 $a = m_i?s$, 则构造迁移

$$pc = q \wedge len_i > 0 \wedge x_{i,0} = s \longrightarrow (v_{i,0}, \dots, v_{i,len_i-1}, len_i, pc) := (v_{i,1}, \dots, v_{i,len_i}, len_i - 1, q').$$

定义 $\Theta = (pc = q_0 \wedge \bigwedge_{i=1}^k (len_i = 0 \wedge \bigwedge_{j=1}^{n_i-1} v_{i,j} = 0))$ 。

定义 $(P)^{sm} = (T, \Theta)$ 。则 $(P)^{sm}$ 是 (B, V) 上的卫式迁移模型。

给定 B 的解释 $I = (D, I_0)$ 其中 $D = N$ 为自然数集合, I_0 将 $\{0, 1, +, -\}$ 解释为自然数上的常数及相应运算, 将 $\Lambda_1 \cup \dots \cup \Lambda_k \cup Q$ 中的元素解释为自然数上的不同常数, 将 $\{>, =, <\}$ 中的符号解释为自然数上的相应关系。

命题 4.20 设 $\zeta: Q \rightarrow D$ 的定义为 $\zeta(q) = I_0(q)$ 。则 $(P)^{sm}$ 与 P 是 ζ 计算等价的。

§4.7 例子

例 4.1 (1) 定义 Σ, S, Δ, I 如下。

$$\begin{aligned}\Sigma &= \{1, 5, lc, tq, qc, qq\}。 \\ S &= \{s_0, \dots, s_6\}。 \\ \Delta &= \{ \\ &\quad (s_0, 1, s_1), (s_0, 5, s_3), (s_1, 1, s_2), (s_1, 5, s_3), (s_1, tq, s_6), \\ &\quad (s_2, tq, s_6), (s_2, lc, s_4), (s_3, tq, s_6), (s_3, lc, s_5), (s_4, qc, s_0), (s_5, qc, s_6), (s_6, qq, s_0) \\ &\quad \}。 \\ I &= \{s_0\}\end{aligned}$$

则 $M_1 = (\Sigma, S, \Delta, I)$ 为标号迁移系统。

若该模型用以表示一个自动售货机的流程设计, 则数字可以理解为投入的钱数, lc 和 tq 分别表示选择物品和退钱, qc 和 qq 分别表示取物品和取钱。

设 $A = \{s_0s_1s_6, s_0s_1s_2s_6, s_0s_1s_2s_4, s_0s_1s_3s_5s_6, s_0s_1s_3s_6, s_0s_3s_5s_6, s_0s_3s_6\}$ 。

则 M_1 的运行集合为 A^ω 。

设 $B = \{1.tq.qq, 1.1.tq.qq, 1.1.lc.qc, 1.5.lc.qc.qq, 1.5.tq.qq, 5.lc.qc.qq, 5.tq.qq\}$ 。

则 M_1 的语言为 B^ω 。

(2) 设 $AP = \{q_0, q_1, q_2, q_3\}$ 。定义 $L: S \rightarrow 2^{AP}$ 如下。

$L(s_0) = \{q_0\}, L(s_1) = \{q_0, q_1\}, L(s_2) = \{q_1\}, L(s_3) = \{q_1\}, L(s_4) = \{q_3\}, L(s_5) = \{q_3\}, L(s_6) = \{q_2\}$ 。

则 $M_2 = (\Sigma, S, \Delta, I, L)$ 为双标号迁移系统。

状态标号 q_0 可用以标识可存钱状态, q_1 可退钱状态, q_2 可取余钱状态, q_3 可取茶状态,

例 4.2 (1) 设 $\Sigma = \{a_1, \dots, a_4, b_1, \dots, b_4\}$ 。定义 $\Delta_0, \Delta_1, \Delta_2, S, \Delta, I$ 如下。

$$\begin{aligned}\Delta_0 &= \{(s_0, a, s_0) \mid a \in \Sigma \setminus \{a_3, b_3\}\} \cup \{(s_0, a_3, s_1), (s_0, b_3, s_2)\} \\ \Delta_1 &= \{(s_1, a, s_1) \mid a \in \Sigma \setminus \{a_3, b_3, a_4\}\} \cup \{(s_1, a_4, s_0)\} \\ \Delta_2 &= \{(s_2, b, s_2) \mid b \in \Sigma \setminus \{a_3, b_3, b_4\}\} \cup \{(s_2, b_4, s_0)\} \\ \hline S &= \{s_0, s_1, s_2\} \\ \Delta &= \Delta_0 \cup \Delta_1 \cup \Delta_2 \\ \hline I &= \{s_0\}\end{aligned}$$

则 $M_1 = (\Sigma, S, \Delta, I)$ 为标号迁移系统。

定义 $X = \{a_1, a_2, a_4, b_1, b_2, b_4\}, Y = \{a_1, a_2, b_1, b_2, b_4\}, Z = \{a_1, a_2, a_4, b_1, b_2\}$ 。

则 M_1 的语言为 $L(M_1) = (X|a_3Y^*a_4|b_3Z^*b_4)^\omega$ 。

(2) 定义 S', Δ', I', F' 如下。

$$S' = S \cup \{s_3\}.$$

$$\Delta' = \Delta \cup \{(s_3, a, s_3) \mid a \in \Sigma\} \cup \{(s_1, a_3, s_3), (s_1, a_4, s_3), (s_2, a_3, s_3), (s_2, a_4, s_3)\}.$$

$$I' = I_0.$$

$$F' = \{s_0, s_1, s_2\}.$$

则 $M_2 = (\Sigma, S', \Delta', I', F')$ 为 Büchi 自动机, 且 $L(M_2) = L(M_1)$ 。

(3) 定义 S'', Δ'', I'', F'' 如下。

$$S'' = \{s_0, s_1\}$$

$$\Delta'' = \{(s_0, a, s_0) \mid a \in \Sigma\} \cup \{(s_1, a, s_1) \mid a \in \Sigma\} \cup \{(s_0, a_3, s_1), (s_0, b_3, s_1)\}$$

$$I'' = \{s_0\}$$

$$F'' = \{s_1\}$$

则 $M_3 = (\Sigma, S'', \Delta'', I'', F'')$ 为 Büchi 自动机, 且其语言为 $L(M_3) = \Sigma^*(a_3|b_3)\Sigma^\omega$ 。

例 4.3 设 $t_1(k, j), \dots, t_8(k, j)$ 和 S, I 如例 2.3 所定义。

(1) 设 $\Sigma = \{a_1, \dots, a_4, b_1, \dots, b_4\}$ 。定义 Δ 为以下迁移集合的并集。

$$\begin{aligned} & \{(s_k, a_1, s_j) \mid t_1(k, j)\} \quad \{(s_k, a_2, s_j) \mid t_2(k, j)\} \\ & \{(s_k, a_3, s_j) \mid t_3(k, j)\} \quad \{(s_k, a_4, s_j) \mid t_4(k, j)\} \\ & \{(s_k, b_1, s_j) \mid t_5(k, j)\} \quad \{(s_k, b_2, s_j) \mid t_6(k, j)\} \\ & \{(s_k, b_3, s_j) \mid t_7(k, j)\} \quad \{(s_k, b_4, s_j) \mid t_8(k, j)\} \end{aligned}$$

则 $M_1 = (\Sigma, S, \Delta, I)$ 为标号迁移系统。

设 M_2, M_3 如例 4.2 所定义。则有 $L(M_1) \subseteq L(M_2)$ 和 $L(M_1) \not\subseteq L(M_3)$ 。

若 M_1 用以表示 2 个进程的互斥算法的带有迁移标识的运行模型, M_2 用以表示一种基于迁移表示的安全性质, M_3 用以表示一种基于迁移表示的必达性质, 则 M_1 具有以上意义的安全性质, 不具有以上意义的必达性质。

(2) 设 F 为以下状态集的集合 (如例 2.3 所定义)。

$$\begin{aligned} & \{s_i \mid \neg(a(i) = 0)\} \\ & \{s_i \mid \neg(a(i) = 1 \wedge (x(i) = 0 \vee t(i) = 0))\} \\ & \{s_i \mid \neg(a(i) = 2)\} \\ & \{s_i \mid \neg(b(i) = 0)\} \\ & \{s_i \mid \neg(b(i) = 1 \wedge (y(i) = 0 \vee t(i) = 1))\} \\ & \{s_i \mid \neg(b(i) = 2)\} \end{aligned}$$

则 $M'_1 = (\Sigma, S, \Delta, I, F)$ 为泛 Büchi 自动机且 $L(M'_1) \subseteq L(M_3)$, 即 M'_1 具有以上意义的必达性质。

例 4.4 (1) 定义 Σ, S, Δ, I, F 如下。

$$\Sigma = \{a, b\}$$

$$S = \{s_0, s_1\}$$

$$\Delta = \{(s_0, a, s_1), (s_0, b, s_0), (s_1, a, s_1), (s_1, b, s_0)\}$$

$$I = \{s_0\}$$

$$F = \{s_0\}$$

则 $B = (\Sigma, S, \Delta, I)$ 为确定型标号迁移系统, $B' = (\Sigma, S, \Delta, I, F)$ 为确定型 Büchi 自动机。

$$L(B') = (a^*b)^\omega.$$

B' 的补语言 $\Sigma^\omega \setminus L(B')$ 为 $\{a, b\}^*a^\omega$ 。由该语言不是任何确定型 Büchi 自动机的语言知确定型 Büchi 自动机不是补封闭的。

(2) 定义 Σ, S, Δ, I, F 如下。

$$\begin{aligned}\Sigma &= \{a, b\} \\ S &= \{s_0, s_1\} \\ \Delta &= \{(s_0, a, s_0), (s_0, b, s_0), (s_0, a, s_1), (s_1, a, s_1)\} \\ I &= \{s_0\} \\ F &= \{s_1\}\end{aligned}$$

则 $A = (\Sigma, S, \Delta', I)$ 为非确定型标号迁移系统, $A' = (\Sigma, S, \Delta', I, F')$ 为非确定型 Büchi 自动机。

$$L(A') = \{a, b\}^*a^\omega = \Sigma^\omega \setminus L(B')$$
 为 B' 的补语言。

例 4.5 设 $t_1(k, j), \dots, t_8(k, j)$ 和 S, I 如例 2.3 所定义。设 $\Sigma = \{a_1, \dots, a_4, b_1, \dots, b_4\}$ 和 $X = \{v, u\}$ 。定义 Δ 为以下迁移集合的并集。

$$\begin{aligned}&\{(s_k, a_1, \{v\}, true, s_j) \mid t_1(k, j)\} \\ &\{(s_k, a_2, \{v\}, v < 2, s_j) \mid t_2(k, j)\} \\ &\{(s_k, a_3, \{v\}, v < 2, s_j) \mid t_3(k, j)\} \\ &\{(s_k, a_4, \{\}, v < 8, s_j) \mid t_4(k, j)\} \\ &\{(s_k, b_1, \{u\}, true, s_j) \mid t_5(k, j)\} \\ &\{(s_k, b_2, \{u\}, u < 1, s_j) \mid t_6(k, j)\} \\ &\{(s_k, b_3, \{u\}, u < 1, s_j) \mid t_7(k, j)\} \\ &\{(s_k, b_4, \{\}, u < 5, s_j) \mid t_8(k, j)\}\end{aligned}$$

则 $M = (\Sigma, S, X, \Delta, I)$ 为时间迁移系统。

定义 A 为以下集合的并集。

$$\begin{aligned}&\{(s_i, [v, u]) \mid a(i) \neq 2 \vee b(i) \neq 2\} \\ &\{(s_i, [v, u]) \mid a(i) = 2 \wedge b(i) \neq 2, v < 8\} \\ &\{(s_i, [v, u]) \mid a(i) \neq 2 \wedge b(i) = 2, u < 5\}\end{aligned}$$

则模型有安全性质 A 。

若该模型用以表示 2 个进程的互斥算法的带有迁移标识和时间约束的运行模型, 则模型具有以上意义的安全性质 (资源的互斥使用且每次使用资源时间合乎给定约束)。

例 4.6 定义 Σ, S, X, Δ, I 如下。

$$\begin{aligned}\Sigma &= \{a, b, c, d\}. \\ X &= \{x, y\}. \\ S &= \{s_0, s_1, s_2, s_3\}. \\ \Delta &= \{(s_0, a, \{x\}, true, s_1), (s_1, b, \{y\}, true, s_2), (s_2, c, \{\}, x < 1, s_3), (s_3, d, \{\}, y > 2, s_0)\}. \\ I &= \{s_0\}.\end{aligned}$$

则 $M = \langle \Sigma, S, X, \Delta, I \rangle$ 为时间迁移系统。

M 的运行集合上的时间字符串集合如下。

$$\{((abcd)^\omega, \tau) \mid \forall j. ((\tau_{4j+3} < \tau_{4j+1} + 1) \wedge (\tau_{4j+4} > \tau_{4j+2} + 2))\}$$

例 4.7 定义 $\Sigma, S, X, \Delta, I, F$ 如下。

$$\Sigma = \{a, b\}.$$

$$S = \{s_0, s_1, s_2, s_3\}.$$

$$X = \{x\}.$$

$$\Delta = \{$$

$$(s_0, a, \{\}, true, s_1), (s_0, a, \{x\}, true, s_2),$$

$$(s_1, b, \{\}, true, s_0), (s_2, b, \{x < 2\}, s_3), (s_3, a, \{x\}, true, s_2)$$

$$\}.$$

$$I = \{s_0\}.$$

$$F = \{s_2\}.$$

则 $M = \langle \Sigma, S, X, \Delta, I, F \rangle$ 为时间自动机。

M 的运行集合上的时间字符串集合为 $\{((ab)^\omega, \tau) \mid \forall j. (\tau_j < \tau_{j+1})\}$ 。

M 的语言为 $\{((ab)^\omega, \tau) \mid \exists i. \forall j \geq i. (\tau_{2j} < \tau_{2j-1} + 2)\}$ 。

例 4.8 定义 $\Sigma, S, X, \Delta, I, F$ 如下。

$$\Sigma = \{a, b\}.$$

$$S = \{s_0, s_1, s_2, s_3\}.$$

$$X = \{x, y\}.$$

$$\Delta = \{ (s_0, a, \{x\}, x = 1, s_1), (s_1, b, \{y\}, true, s_2), (s_2, a, \{x\}, x = 1, s_3), (s_3, b, \{y\}, y < 1, s_2) \}.$$

$$I = \{s_0\}.$$

$$F = \{s_2\}.$$

则 $M = \langle \Sigma, S, X, \Delta, I, F \rangle$ 为时间自动机。

M 的运行的集合上的时间字符串集合与其语言如下。

$$\{((ab)^\omega, \tau) \mid \forall j. ((\tau_{2j-1} = j) \wedge (\tau_{2j} - \tau_{2j-1} > \tau_{2j+2} - \tau_{2j+1}))\}$$

我们有 $\lim_{j \rightarrow \infty} (\tau_{j+2} - \tau_j) = 1$ 。

例 4.9 定义 $\Sigma, S, X, \Delta, I, F$ 如下。

$$\begin{aligned}
\Sigma &= \{a, b\}。 \\
S &= \{gr, gr2, yr, rr, rg, rg2\}。 \\
X &= \{u\}。 \\
\Delta &= \{ \\
&\quad (gr, a, \{u\}, true, gr2), \quad (gr2, b, \{u\}, u = 5, yr), \\
&\quad (yr, b, \{u\}, u = 1, rr), \quad (rr, b, \{u\}, u = 1, rg), \\
&\quad (rg, a, \{u\}, u < 6, rg2), \quad (rg, b, \{u\}, u = 6, gr), \quad (rg2, b, \{u\}, u = 6, gr) \\
&\quad \} \\
I &= \{gr\}。 \\
F &= \{rg\}。
\end{aligned}$$

则 $M = \langle \Sigma, S, X, \Delta, I, F \rangle$ 为时间自动机。

若该模型用以表示一种行人过街用红绿灯（行人方向有红绿灯，汽车有红绿灯黄灯）的流程设计，则初始状态 gr 可理解为汽车绿灯(g)和行人红灯(r)， a 理解为行人按过街按钮， b 理解为为系统内部动作，数字理解为等待时间。

例 4.10 定义 $\lambda_0, \lambda_1 : R \rightarrow R$ 如下： $\lambda_0(x) = x + 1$ 且 $\lambda_1(x) = x - 1$ 。定义 $\Sigma, S, X, \Delta, I, flow$ 如下。

$$\begin{aligned}
\Sigma &= \{a, b\}。 \\
S &= \{s_0, s_1\}。 \\
X &= \{x\}。 \\
\Delta &= \{(s_0, a, \lambda_0, x = 900, s_1), (s_1, b, \lambda_1, x = 100, s_0)\}。 \\
I &= \{(s_0, [0])\}。 \\
flow(s_0) &= (\dot{x} = 2), \quad flow(s_1) = (\dot{x} = -1)。
\end{aligned}$$

则 $M = \langle \Sigma, S, X, \Delta, I, flow \rangle$ 为混成迁移系统。

定义 $A = \{(s_i, [x]) \mid x < 1000, i = 0 \vee i = 1\}$ 。则模型有安全性质 A 。

若该模型用以表示一种水箱容量 1000 的水量控制模型，则模型具有以上意义的安全性质（不会出现水满溢出）。

例 4.11 设 $\langle \Lambda, n \rangle = \langle \{0, 1, 2, 3\}, 2 \rangle$ 为通道类型。定义 $Q_1, M_1, \Delta_1, Q_2, M_2, \Delta_2$ 如下。

$$\begin{aligned}
Q_1 &= \{s_0, s_1, s_2, s_3\} \\
M_1 &= \{m_1, m_2 \mid m_1, m_2 \in \langle \Lambda, n \rangle\} \\
\Delta_1 &= \{ \\
&\quad (s_0, m_1!1, s_0), (s_0, m_1!2, s_0), (s_0, m_2?0, s_0), (s_0, m_2?1, s_1), \\
&\quad (s_0, m_2?2, s_2), (s_0, m_2?3, s_3), (s_1, m_1!2, s_1), (s_1, m_1!3, s_1), \\
&\quad (s_1, m_2?0, s_0), (s_1, m_2?1, s_1), (s_1, m_2?2, s_2), (s_1, m_2?3, s_3), \\
&\quad (s_2, m_1!3, s_2), (s_2, m_1!0, s_2), (s_2, m_2?0, s_0), (s_2, m_2?1, s_1), \\
&\quad (s_2, m_2?2, s_2), (s_2, m_2?3, s_3), (s_3, m_1!0, s_3), (s_3, m_1!1, s_3), \\
&\quad (s_3, m_2?0, s_0), (s_3, m_2?1, s_1), (s_3, m_2?2, s_2), (s_3, m_2?3, s_3) \\
&\quad \}. \\
Q_2 &= \{t_0, t_1, t_2, t_3\} \\
M_2 &= \{m_1, m_2 \mid m_1, m_2 \in \langle \Lambda, n \rangle\} \\
\Delta_2 &= \{ \\
&\quad (t_0, m_2!0, t_0), (t_0, m_1?0, t_0), (t_0, m_1?1, t_1), (t_0, m_1?2, t_0), (t_0, m_1?3, t_0), \\
&\quad (t_1, m_2!1, t_1), (t_1, m_1?0, t_1), (t_1, m_1?1, t_1), (t_1, m_1?2, t_2), (t_1, m_1?3, t_1), \\
&\quad (t_2, m_2!2, t_2), (t_2, m_1?0, t_2), (t_2, m_1?1, t_2), (t_2, m_1?2, t_2), (t_2, m_1?3, t_3), \\
&\quad (t_3, m_2!3, t_3), (t_3, m_1?0, t_0), (t_3, m_1?1, t_3), (t_3, m_1?2, t_3), (t_3, m_1?3, t_3) \\
&\quad \}
\end{aligned}$$

则 $A = \langle Q_1, M_1, \Delta_1, s_0 \rangle$ 和 $B = \langle Q_2, M_2, \Delta_2, t_0 \rangle$ 为通信单元。

通信系统 $\{A, B\}$ 是由这两个通信单元组成的可用以表示窗口长度为2的一种简化的窗口协议的通信系统模型。 A 所在的状态表示 A 将发送序号为状态下标加1或2的信息， B 所在的状态表示 B 需要接收的信息为状态下标加1的信息。

例 4.12 有限状态变量的通信单元表示。假定有限状态变量 m 的值域为 $\{0, 1, 2\}$ 。构造通信单元 $B = \langle Q, M, \Delta, q_0 \rangle$ 如下。

$$\begin{aligned}
Q &= \{q_0, q_1, q_2\} \\
M &= \{mi, mo \mid mi, mo \in \langle \{0, 1, 2, r\}, 1 \rangle\} \\
\Delta &= \{ \\
&\quad (q_0, mi?0, q_0), (q_0, mi?1, q_1), (q_0, mi?2, q_2), (q_0, mi?r, r_0), (r_0, mo!0, q_0), \\
&\quad (q_1, mi?0, q_0), (q_1, mi?1, q_1), (q_1, mi?2, q_2), (q_1, mi?r, r_1), (r_1, mo!1, q_1), \\
&\quad (q_2, mi?0, q_0), (q_2, mi?1, q_1), (q_2, mi?2, q_2), (q_2, mi?r, r_2), (r_2, mo!2, q_2) \\
&\quad \}
\end{aligned}$$

从状态 q 经过 $m = i$ 赋值语句 (其中 $i \in \{0, 1, 2\}$ 为常量) 转到状态 q' 的状态迁移对应于 $(q, mi!i, q')$ 。

设另有一变量 a 。类似地， a 可用使用相应的 ai, ao 通道的通信单元表示。

从状态 q 经过 $a = m$ 赋值语句转到状态 q' 的状态迁移对应于以下迁移的集合。

$$\begin{aligned}
&(q, mi!r, p_1), \\
&(p_1, mo?0, p_{10}), (p_1, mo?1, p_{11}), (p_1, mo?2, p_{12}), \\
&(p_{10}, ai!0, q'), (p_{11}, ai!1, q'), (p_{12}, ai!2, q')
\end{aligned}$$

从状态 q 根据 a 是否为 m 的条件语句转到状态 q' 或 q^* 的状态迁移对应于以下迁移的集合。

$$\begin{aligned} &(q, a!r, p_1), (p_1, m!r, p_2), \\ &(p_2, a0?0, p_{20}), (p_2, a0?1, p_{21}), (p_2, a0?2, p_{22}), \\ &(p_{20}, m0?0, q'), (p_{20}, m0?1, q^*), (p_{20}, m0?2, q^*), \\ &(p_{21}, m0?1, q'), (p_{21}, m0?0, q^*), (p_{21}, m0?2, q^*), \\ &(p_{22}, m0?2, q'), (p_{22}, m0?0, q^*), (p_{22}, m0?1, q^*) \end{aligned}$$

例 4.13 定义 P, T, F, m_0 如下。

$$P = \{u_0, u_1, u_2, u_3, v_0, v_1, v_2, v_3, uv\}$$

$$T = \{s_0, s_1, s_2, s_3, t_0, t_1, t_2, t_3\}$$

F 由以下 $\circ p()$ 和 $p^\circ()$ 的定义构造。

$$\circ p(s_0) = \{u_0\} \quad \circ p(s_1) = \{u_1, uv\} \quad \circ p(s_2) = \{u_2\} \quad \circ p(s_3) = \{u_3\}$$

$$\circ p(t_0) = \{v_0\} \quad \circ p(t_1) = \{v_1, uv\} \quad \circ p(t_2) = \{v_2\} \quad \circ p(t_3) = \{v_3\}$$

$$p^\circ(s_0) = \{u_1\} \quad p^\circ(s_1) = \{u_2\} \quad p^\circ(s_2) = \{u_3, uv\} \quad p^\circ(s_3) = \{u_1\}$$

$$p^\circ(t_0) = \{v_0\} \quad p^\circ(t_1) = \{v_2\} \quad p^\circ(t_2) = \{v_3, uv\} \quad p^\circ(t_3) = \{v_1\}$$

$$m_0(p) = 1 \text{ 若 } p \in \{u_0, v_0, uv\} \text{ 且 } m_0(p) = 0 \text{ 若 } p \notin \{u_0, v_0, uv\}.$$

则 $\langle P, T, F, m_0 \rangle$ 为Petri网模型。

定义 A 为满足以下性质的 m 的集合： $m(u_2) = m(v_2) = 1$ 。则 A 不是模型的可达性质。

若该模型用以表示一个资源共享和互斥的抽象模型，则具有以上意义的安全（互斥）性质。

例 4.14 定义 P, T, F, m_0 如下。

$$P = \{u_0, u_1, u_2, v_0, v_1, v_2\}$$

$$T = \{s_0, s_1, s_2, t_0, t_1, t_2\}$$

F 由以下 $\circ p()$ 和 $p^\circ()$ 的定义构造。

$$\circ p(s_0) = \{u_0\} \quad \circ p(s_1) = \{u_1, v_0\} \quad \circ p(s_2) = \{u_2\}$$

$$\circ p(t_0) = \{v_0\} \quad \circ p(t_1) = \{u_0, v_1\} \quad \circ p(t_2) = \{u_2\}$$

$$p^\circ(s_0) = \{u_1\} \quad p^\circ(s_1) = \{u_2\} \quad p^\circ(s_2) = \{u_0, v_0\}$$

$$p^\circ(t_0) = \{u_0\} \quad p^\circ(t_1) = \{v_2\} \quad p^\circ(t_2) = \{u_0, v_0\}$$

$$m_0(p) = 1 \text{ 若 } p \in \{u_0, v_0\} \text{ 且 } m_0(p) = 0 \text{ 若 } p \notin \{u_0, v_0\}.$$

则 $\langle P, T, F, m_0 \rangle$ 为Petri网模型。

定义 m 如下： $m(p) = 1$ 若 $p \in \{u_1, v_1\}$ 且 $m(p) = 0$ 若 $p \notin \{u_1, v_1\}$ 。则 m 为可达的死锁状态。

若该模型用以表示一个互相合作的抽象模型，则存在以上意义的死锁问题。

§4.8 练习

1. 给定泛Büchi 自动机 $\mathcal{A} = \langle \Sigma, S, \Delta, I, F \rangle$ 。构造Büchi 自动机 $\mathcal{B} = \langle \Sigma, S', \Delta', I', F' \rangle$ 使得 $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{A})$ 。
2. 定义泛Büchi 自动机的交运算和并运算，即给定泛Büchi 自动机 $\mathcal{B}_1 = \langle \Sigma, S_1, \Delta_1, I_1, F_1 \rangle$ 和 $\mathcal{B}_2 = \langle \Sigma, S_2, \Delta_2, I_2, F_2 \rangle$ ，定义泛Büchi 自动机 $\mathcal{B} = \mathcal{B}_1 \cap \mathcal{B}_2$ 和 $\mathcal{B}' = \mathcal{B}_1 \cup \mathcal{B}_2$ 使得 $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{B}_1) \cap \mathcal{L}(\mathcal{B}_2)$ 和 $\mathcal{L}(\mathcal{B}') = \mathcal{L}(\mathcal{B}_1) \cup \mathcal{L}(\mathcal{B}_2)$ 。

3. 用时间自动机描述交通灯的变化。要求信号灯的周期性变化次序为绿黄红，且黄灯的长度为1，绿灯的长度不小于10，红灯的长度不小于10，变化周期绿黄红的总长度为30。试改进模型使得若绿红两灯的长度差别不为零则长度差别逐渐减少。
4. 设有四个生产者A、B、C、D和一个消费者E。用库所及其赋值表示生产者和消费者，用迁移表示生产和消费，用库所及其赋值表示产品类型和数量。用Petri网描述以下过程。A不停地生产零件a，B不停地生产零件b，C使用a和b生产零件c，D使用a和c生产产品d，E不停地消费产品d。
5. 定义一种通信单元与标号Kripke结构的合理的等价关系和一种满足所定义等价关系的由通信单元构造标号Kripke结构的算法。

§5 线性时序逻辑

在命题逻辑和一阶逻辑的基础上增加模态算子用以描述时间先后次序有关的性质的一类逻辑称为时序逻辑。本章介绍线性时序逻辑。

§5.1 命题线性时序逻辑(PLTL)

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 。用 p 表示 AP 中的任意命题。PLTL公式的集合由以下语法给出。

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \rightarrow \psi \mid \phi \leftrightarrow \psi \mid O\phi \mid \diamond\phi \mid \square\phi \mid \phi U \psi \mid \phi R \psi$$

符号 $O, \diamond, \square, U, R$ 称为PLTL的时序算子。前三个时序算子在不同的场合和应用背景下也写成 X, F, G 。

模型： AP 上的PLTL公式在 AP 上的标号Kripke结构上解释。

PLTL公式的语义： 给定 $M = \langle S, R, I, L \rangle$ 。状态 $s \in S$ 满足PLTL公式 φ 记为 $M, s \models \varphi$ 当且仅当对所有 s 起点的无穷路径 π 有 $M, \pi \models \varphi$ 。用 π^k 来表示 π_k 开始的序列 $[\pi_i]_{i \geq k}$ 。 $M, \pi \models \varphi$ 定义如下。

$M, \pi \models p$	若 $p \in AP$ 且 $p \in L(\pi_0)$
$M, \pi \models \neg\varphi$	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 $M, \pi \models \psi$
$M, \pi \models \varphi \rightarrow \psi$	若 $M, \pi \models \varphi$ 则 $M, \pi \models \psi$
$M, \pi \models \varphi \leftrightarrow \psi$	若 $M, \pi \models \varphi \rightarrow \psi$ 且 $M, \pi \models \psi \rightarrow \varphi$
$M, \pi \models O\varphi$	若 $M, \pi^1 \models \varphi$
$M, \pi \models \square\psi$	若 $\forall i \geq 0, M, \pi^i \models \psi$
$M, \pi \models \diamond\varphi$	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 $\forall i \geq 0. (M, \pi^i \models \psi$ 或 $\exists 0 \leq j < i, M, \pi^j \models \varphi)$

定义 5.1 $M \models \varphi$ 当且仅当对所有 $s \in I$ 有 $M, s \models \varphi$ 。

可满足性： 设 φ 为PLTL公式。 φ 是可满足的，当且仅当存在标号Kripke结构 M 使得 $M \models \varphi$ 。

重言式和等价： 设 φ 和 ψ 为PLTL公式。 φ 是重言式，记作 $\models \varphi$ ，当且仅当对任意标号Kripke结构 M 有 $M \models \varphi$ 。 φ 等价于 ψ ，记作 $\varphi \equiv \psi$ ，当且仅当 $\models \varphi \leftrightarrow \psi$ 。

等价公式： 为方便起见，定义 $\top = (p_0 \vee \neg p_0)$ 和 $\perp = (p_0 \wedge \neg p_0)$ 其中 $p_0 \in AP$ 为给定命题。设 φ 和 ψ 为PLTL公式。我们有以下等价的公式对。

1. $O\varphi \equiv \neg(O\neg\varphi)$
2. $\Box\varphi \equiv \neg(\Diamond\neg\varphi)$
3. $\varphi R\psi \equiv \neg(\neg\varphi U\neg\psi)$
4. $\Box\varphi \equiv (\varphi \wedge O(\Box\varphi))$
5. $\Diamond\varphi \equiv (\varphi \vee O(\Diamond\varphi))$
6. $\varphi R\psi \equiv (\psi \wedge (\varphi \vee O(\varphi R\psi)))$
7. $\varphi U\psi \equiv (\psi \vee (\varphi \wedge O(\varphi U\psi)))$
8. $\Diamond\varphi \equiv (\top U\varphi)$
9. $\Box\varphi \equiv (\perp R\varphi)$
10. $\varphi R\psi \equiv (\psi U(\varphi \wedge \psi) \vee G\psi)$

完全集： 一个命题逻辑联接符和时序算子的集合 Y 称为PLTL的完全集，当且仅当每个PLTL公式都等价于一个命题逻辑联接符和时序算子都在 Y 中的PLTL公式。一个时序算子的集合 Y 称为PLTL的时序算子完全集，当且仅当每个PLTL公式都等价于一个时序算子在 Y 中的PLTL公式。

极小完全集： 一个命题逻辑联接符和时序算子的集合 Y 称为PLTL的极小完全集，当且仅当 Y 是PLTL的完全集且 Y 的真子集都不是PLTL的完全集。一个时序算子的集合 Y 称为PLTL的时序算子极小完全集，当且仅当 Y 是PLTL的时序算子完全集且 Y 的真子集都不是PLTL的时序算子完全集。

命题 5.1 $\{O, U\}$ 是PLTL的时序算子极小完全集。

由以上等价的公式类型，我们知道 $\{O, U\}$ 是PLTL的时序算子完全集。设 $AP = \{p\}$ 和 $\varphi = (\neg p U p)$ 。我们可以构造两个序列的标号Kripke结构 M_k 和 M'_k 使得 $M_k \models \varphi$ 和 $M'_k \not\models \varphi$ ，但不存在任何只使用时序算子 O 的公式具有同样性质，即 $(\neg p U p)$ 不等价于任何一个仅包含 O 和命题逻辑联接符的PLTL公式，因而 $\{O\}$ 不是PLTL的时序算子完全集。类似地，我们可以证明 $\{U\}$ 不是PLTL的时序算子完全集。

表达能力的强弱关系： 用 $LTL(Y)$ 表示仅用集合 Y 中的时序算子和命题逻辑连接符构造的公式的集合。用 $>$ 表示表达能力存在强弱关系。则有如下结论。

- $LTL(\{O, U\}) > LTL(\{O, F\}) > LTL(\{O\})$ 。
- $LTL(\{O, U\}) > LTL(\{U\}) > LTL(\{F\})$ 。
- $LTL(\{O, F\}) > LTL(\{F\})$ 。

对于 $\{O, U, F\}$ 的不同子集，表达能力的强弱关系若不能由以上关系推导得出，则这些子集的表达能力的存在互补关系。

NNF范式： 只使用逻辑连接符 \wedge, \vee, \neg 且逻辑联接符 \neg 只出现在命题前面的公式称为NNF范式。

命题 5.2 每个PLTL公式等价于一个PLTL的NNF范式公式。

对于每个PLTL公式，我们首先可以将其转换成等价的只使用 $\{\neg, \vee, \wedge\}$ 中命题逻辑联接符和 $\{O, U\}$ 中时序算子的公式。然后使用以下等价关系将其转换成时序算子在 $\{O, U, R\}$ 中的NNF范式公式。

$$\begin{array}{l}
 \hline
 \neg\neg\varphi \quad \equiv \quad \varphi \\
 \neg(\varphi \vee \psi) \quad \equiv \quad \neg\varphi \wedge \neg\psi \\
 \neg(\varphi \wedge \psi) \quad \equiv \quad \neg\varphi \vee \neg\psi \\
 \hline
 \neg(O\varphi) \quad \equiv \quad O(\neg\varphi) \\
 \neg(\varphi U\psi) \quad \equiv \quad \neg\varphi R\neg\psi \\
 \neg(\varphi R\psi) \quad \equiv \quad \neg\varphi U\neg\psi \\
 \hline
 \end{array}$$

可满足性判定问题： 判定PLTL公式是否可满足的问题称为PLTL可满足性问题。PLTL可满足性问题的复杂性为PSPACE完全。只允许时序算子 \square 和 \diamond 的PLTL公式的子集记为PLTL(F)。PLTL(F)可满足性问题的复杂性为NP完全。

命题 5.3 设 φ 为PLTL公式。定义 $|\varphi|$ 为 φ 中出现的符号个数。若 φ 可满足，则存在 M 和 $\pi = \pi_0 \cdots \pi_{i-1}(\pi_i \cdots \pi_k)^\omega$ 使得 $M, \pi \models \varphi$ 且 $k \leq 2^{|\varphi|}$ 。

模型检测问题： 判定Kripke模型是否满足PLTL公式的问题称为PLTL模型检测问题。PLTL模型检测问题的复杂性为PSPACE完全。

§5.1.1 PLTL公式的推理

以 $\{O, U, \square, \diamond\}$ 为PLTL公式的时序算子集。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推理系统；第三部分为时序推理规则。设 φ 和 ψ 为PLTL公式。

时序逻辑公理： 我们有以下公理。

$$\begin{array}{l}
 \hline
 \diamond\neg\neg\varphi \leftrightarrow \diamond\varphi \qquad \square(\varphi \rightarrow O\varphi) \rightarrow (\varphi \rightarrow \square\varphi) \\
 \square(\varphi \rightarrow \psi) \rightarrow (\square\varphi \rightarrow \square\psi) \quad O\neg\varphi \leftrightarrow \neg O\varphi \\
 \square\varphi \rightarrow \varphi \qquad O(\varphi \rightarrow \psi) \rightarrow (O\varphi \rightarrow O\psi) \\
 \square\varphi \rightarrow O\varphi \qquad (\varphi U\psi) \leftrightarrow (\psi \vee (\varphi \wedge O(\varphi U\psi))) \\
 \square\varphi \rightarrow O\square\varphi \qquad (\varphi U\psi) \rightarrow \diamond\psi \\
 \hline
 \end{array}$$

命题逻辑推理系统：

(AX) 若 φ 是命题逻辑的重言式的实例，则 $\vdash \varphi$ 。

(MP) 若 $\vdash \varphi \rightarrow \psi$ 且 $\vdash \varphi$ ，则 $\vdash \psi$ 。

时序推理规则:

(G) 若 $\vdash \varphi$, 则 $\vdash \Box \varphi$.

命题 5.4 *PLTL*推理系统是可靠且完备的。

§5.1.2 *PLTL*限界语义

给定有穷状态标号Kripke结构 $M = \langle S, R, I, L \rangle$ 。称长度 $k + 1$ 的路径为 k 路径。 k 路径 $[\pi_i]_{i=0}^k$ 称为 (k, ℓ) 环当且仅当 $R(\pi_k, \pi_\ell)$ 。

定义 5.2 设 φ 为*PLTL*公式。 $M, s \models_E \varphi$ 当且仅当存在 s 起点的无穷路径 π 使得 $M, \pi \models \varphi$ 。

命题 5.5 设 φ 为*PLTL*公式。 $M, s \models_E \varphi$ 当且仅当存在一个 s 起点的 (k, ℓ) 环使得以下成立:

$$M, \pi_0 \cdots \pi_{\ell-1} (\pi_\ell \cdots \pi_k)^\omega \models \varphi \text{ 且 } k \leq |M| \cdot 2^{|\varphi|}.$$

定义 5.3 设 φ 为*PLTL*公式。 $M \models_E \varphi$ 当且仅当存在 $s \in I$ 使得 $M, s \models \varphi$ 。

推论 5.1 设 φ 为*PLTL*公式。 $M \not\models \varphi$ 当且仅当 $M \models_E \neg \varphi$ 。

以下考虑仅用 O, \Box, U 时序算子的NNF范式的*PLTL*公式。

设 π 为 k 路径。用 $U_{x,y}(\pi, \varphi, \psi, i, n)$ 表示以下公式（用于以下非环和环的语义中 $\models_{x,y}^i$ 的定义）。

$$U_{x,y}(\pi, \varphi, \psi, i, n) = \bigvee_{j=i}^n ((M, \pi \models_{x,y}^j \psi) \wedge \bigwedge_{m=i}^{j-1} (M, \pi \models_{x,y}^m \varphi))$$

非环的语义: 定义 $M, \pi \models_{a,k} \varphi$ 为 $M, \pi \models_{a,k}^0 \varphi$ 且后者的定义如下。

$M, \pi \models_{a,k}^i p$	若 $p \in AP$ 且 $p \in L(\pi_i)$
$M, \pi \models_{a,k}^i \neg p$	若 $M, \pi \not\models_{a,k}^i p$
$M, \pi \models_{a,k}^i \varphi \wedge \psi$	若 $M, \pi \models_{a,k}^i \varphi$ 且 $M, \pi \models_{a,k}^i \psi$
$M, \pi \models_{a,k}^i \varphi \vee \psi$	若 $M, \pi \models_{a,k}^i \varphi$ 或 $M, \pi \models_{a,k}^i \psi$
$M, \pi \models_{a,k}^i O\varphi$	若 $i < k$ 且 $M, \pi \models_{a,k}^{i+1} \varphi$
$M, \pi \models_{a,k}^i \Box \varphi$	若 $false$
$M, \pi \models_{a,k}^i \varphi U \psi$	若 $U_{a,k}(\pi, \varphi, \psi, i, k)$

命题 5.6 $M, \pi \models_{a,k} \varphi$ 当且仅当对任意 $\pi' \in \pi \cdot (S)^\omega$, $M, \pi' \models \varphi$ 。

环的语义: 设 $0 \leq \ell \leq k$ 。定义 $M, \pi \models_{\ell,k} \varphi$ 为 $M, \pi \models_{\ell,k}^0 \varphi$ 且后者的定义如下。

$M, \pi \models_{\ell,k}^i p$	若 $p \in AP$ 且 $p \in L(\pi_i)$
$M, \pi \models_{\ell,k}^i \neg p$	若 $M, \pi \not\models_{\ell,k}^i p$
$M, \pi \models_{\ell,k}^i \varphi \wedge \psi$	若 $M, \pi \models_{\ell,k}^i \varphi$ 且 $M, \pi \models_{\ell,k}^i \psi$
$M, \pi \models_{\ell,k}^i \varphi \vee \psi$	若 $M, \pi \models_{\ell,k}^i \varphi$ 或 $M, \pi \models_{\ell,k}^i \psi$
$M, \pi \models_{\ell,k}^i O\varphi$	若 $i < k$ 且 $M, \pi \models_{\ell,k}^{i+1} \varphi$ 或 $i = k$ 且 $M, \pi \models_{\ell,k}^\ell \varphi$
$M, \pi \models_{\ell,k}^i \Box \varphi$	若 $\bigwedge_{j=\min(i,\ell)}^k (M, \pi \models_{\ell,k}^j \varphi)$
$M, \pi \models_{\ell,k}^i \varphi U \psi$	若 $U_{\ell,k}(\pi, \varphi, \psi, i, k) \vee \bigwedge_{j=i}^k (M, \pi \models_{\ell,k}^j \varphi) \wedge U_{\ell,k}(\pi, \varphi, \psi, \ell, i-1)$

命题 5.7 $M, \pi \models_{\ell,k} \varphi$ 其中 $0 \leq \ell \leq k$, 当且仅当 $M, \pi_0 \cdots \pi_{\ell-1} (\pi_\ell \cdots \pi_k)^\omega \models \varphi$ 。

限界语义:

定义 5.4 $M, \pi \models_k \varphi$ 当且仅当 $M, \pi \models_{a,k} \varphi$ 或 $\bigvee_{\ell=0}^k (R(\pi_k, \pi_\ell) \wedge (M, \pi \models_{\ell,k} \varphi))$ 。

用 $M, s \models_{E,k} \varphi$ 表示存在 s 起点的 k 路径 π 使得 $M, \pi \models_k \varphi$ 。

命题 5.8 对所有 $k \geq 0$, 若 $M, s \models_{E,k} \varphi$, 则 $M, s \models_{E,k+1} \varphi$ 。

正确性与完备性:

命题 5.9 $M, s \models_E \varphi$ 当且仅当存在 $k \geq 0$ 使得 $M, s \models_{E,k} \varphi$ 。

用 $M \models_{E,k} \varphi$ 表示存在初始状态为起点的 k 路径 π 使得 $M, \pi \models_k \varphi$ 。

推论 5.2 $M \models_E \varphi$ 当且仅当存在 $k \geq 0$ 使得 $M \models_{E,k} \varphi$ 。

推论 5.3 $M \not\models \varphi$ 当且仅当存在 $k \geq 0$ 使得 $M \models_{E,k} \neg \varphi$ 。

完备阈值: 定义 (M, φ) 的完备阈值为满足以下条件的 k 。

$$M \not\models_{E,k} \neg \varphi, \text{ 则对所有 } i \geq 0, M \not\models_{E,k+i} \neg \varphi.$$

记 (M, φ) 的最小完备阈值为 $lct(M, \varphi)$ 。

对任何 M 和 φ , 最小完备阈值 $lct(M, \varphi)$ 存在且不超过 $|M| \cdot 2^{|\varphi|}$ 。同时我们有以下结论。

推论 5.4 若 $k \geq lct(M, \varphi)$ 且 $M \not\models_{E,k} \neg \varphi$, 则 $M \models \varphi$ 。

推论 5.5 若 $M \models \varphi$, 则 $lct(M, \varphi) = 0$ 。

注: 若不考虑非环语义部分, 以上理论的各个命题同样成立, 但对给定 (M, φ) , 其最小完备阈值可能不同。

§5.1.3 PLTL公式的Büchi 自动机表示

记 $L(\pi) = [L(\pi_i)]_{i \geq 0}$ 。

2^{AP} 上的字符串 $\xi \in (2^{AP})^\omega$ 满足 φ 当且仅当存在 $M = \langle S, R, I, L \rangle$ 和 M 的计算 π 使得 $L(\pi) = \xi$ 且 $M, \pi \models \varphi$ 。

记 $[[\varphi]] \subseteq (2^{AP})^\omega$ 为满足 φ 的字符串的集合。

命题 5.10 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 Büchi 自动机可通过泛 Büchi 自动机的转换得到, 以下是一个构造语言等价于 $[[\varphi]]$ 的泛 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 。

(a) 化简: 对每个新构造的状态 $y = (b, c, d)$, 若 c 中命题的集合是不可满足的, 则从 S 中删除 y ; 若 $c \cap b \neq \emptyset$ (广义上讲, 若 $\varphi \in b$ 且 c 蕴含 φ , 则可认为 $\varphi \in c \cap b$), 则从 b 中删除公共部分 $c \cap b$ 。

(b) 检查是否重复构造: 对每个新构造的状态 y , 若 $y \in S$ (若 $y' \in S$ 且 y' 与 y 要满足的公式是等价的, 则亦可认为 $y = y' \in S$) 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; 若 $y \notin S$, 记 $a(y) = a(x)$, 并将 y 加入状态集 S 。

(c) 完成替代: 修改 $a(x) = \emptyset$ 表示 x 已为新构造的状态所替代 (并记录 x 与这些状态的关系待用); 返回第二步直至没有新加入状态集的状态。

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 $\forall p \in AP. (\{p, \neg p\} \not\subseteq \sigma \cup c') \wedge (p \in c' \rightarrow p \in \sigma)$ 。

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 $f_\varphi = \{(b, c, d) \in S \mid \varphi \in c \rightarrow \varphi_1 \in c\}$, 并将其加入接受状态集集合 F 。

命题 5.11 给定 *PLTL* 公式 φ 。设 a 为 φ 中出现的 U 和 R 的个数。定义 $B_\varphi = \langle 2^{AP}, S, \Delta, I, F \rangle$ 其中 S, Δ, I, F 的构造方法如上。则 $\mathcal{L}(B_\varphi) = [[\varphi]]$ 且 $|S| \leq 2^{|\varphi| - a}$ 。

推论 5.6 给定 *PLTL* 公式 φ 。 φ 可满足当且仅当 $\mathcal{L}(B_\varphi)$ 非空。

§5.2 *PLTL* 公式的不动点表示与线性 μ 演算 (ν TL)

线性结构: 给定原子命题集合 AP 。一个线性结构为三元组 $\langle S, \zeta, L \rangle$, 其中 S 为状态集合, $\zeta = [\zeta_i]_{i \geq 0} \in S^\omega$ 为 S 的无穷序列, $L: S \rightarrow 2^{AP}$ 为标号函数。

定义 5.5 给定线性结构 $\langle S, \zeta, L \rangle$ 和 *PLTL* 公式 φ 。若存在标号 *Kripke* 模型 $M = \langle S, R, I, L \rangle$ 使得 ζ 是 M 的计算且 $M, \zeta \models \varphi$, 则称 $\langle S, \zeta, L \rangle \models \varphi$ 。

将 \mathbf{N} 的子集 $\{i \mid \langle S, \zeta^i, L \rangle \models \varphi\}$ 记为 $[[\varphi]]$ 。定义函数 $o: 2^{\mathbf{N}} \rightarrow 2^{\mathbf{N}}$ 如下:

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

我们有以下等式:

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

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

$$[[\varphi U \psi]] = \mu Z. ([[\psi]]) \cup ([[\varphi]]) \cap o(Z)$$

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

$\diamond\varphi$	$= \mu Z. (\varphi \vee OZ)$
$\square\varphi$	$= \nu Z. (\varphi \wedge OZ)$
$\varphi U \psi$	$= \mu Z. (\psi \vee (\varphi \wedge OZ))$
$\varphi R \psi$	$= \nu Z. (\psi \wedge (\varphi \vee OZ))$

命题 5.12 给定线性结构 $\langle S, \zeta, L \rangle$ 和PLTL公式 φ 。 $\langle S, \zeta, L \rangle \models \varphi$ 当且仅当 $0 \in [[\varphi]]$ 。

§5.2.1 线性 μ 演算：

由以上讨论知PLTL公式可以用时序算子 O 和不动点表示。在这种表示中，命题变量只出现在 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 ν TL。给定一个原子命题集合 AP ，一个变量集合 V 。用 p 表示 AP 中任意命题， X 表示 V 中任意变量。 ν TL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 $\mu X. \phi$ 和 $\nu X. \phi$ 的 ϕ 中不受 \neg 的 X 必须在偶数个 \neg 符号的作用范围之下。

ν TL公式在线性结构上的语义解释： 给定线性结构 $\langle S, \zeta, L \rangle$ 和 ν TL公式 φ 。设 $e : V \rightarrow 2^N$ 为变量到 N 子集的赋值。 ν TL公式的语义如下：

$[[p]]e$	$= \{i \in N \mid p \in L(\zeta_i)\}$
$[[X]]e$	$= e(X)$
$[[\neg\phi]]e$	$= N \setminus [[\phi]]e$
$[[\phi_1 \wedge \phi_2]]e$	$= [[\phi_1]]e \cap [[\phi_2]]e$
$[[\phi_1 \vee \phi_2]]e$	$= [[\phi_1]]e \cup [[\phi_2]]e$
$[[O\phi]]e$	$= \{i \in N \mid i + 1 \in [[\phi]]e\}$
$[[\mu X. \phi]]e$	$= \cap \{M \subseteq N \mid [[\phi]]e[X/M] \subseteq M\}$
$[[\nu X. \phi]]e$	$= \cup \{M \subseteq N \mid M \subseteq [[\phi]]e[X/M]\}$

定义 5.6 给定线性结构 $\langle S, \zeta, L \rangle$ 和 ν TL公式 φ 。 $\langle S, \zeta, L \rangle \models \varphi$ 当且仅当 $0 \in [[\varphi]]$ 。

可满足性： 设 φ 为 ν TL公式。 φ 是可满足的，当且仅当存在 $\langle S, \zeta, L \rangle$ 使得 $\langle S, \zeta, L \rangle \models \varphi$ 。

重言式和等价： 设 φ 为 ν TL公式。 φ 是重言式，记作 $\models \varphi$ ，当且仅当对任意 $\langle S, \zeta, L \rangle$ 有 $\langle S, \zeta, L \rangle \models \varphi$ 。 φ 等价于 ψ ，记作 $\varphi \equiv \psi$ ，当且仅当 $\models (\varphi \wedge \psi) \vee (\neg \varphi \wedge \neg \psi)$ 。

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 ν TL公式等价于一个 ν TL的NNF范式。一个 ν TL的NNF范式可应用以下对偶关系构造。

$$\begin{array}{l} \hline \neg(O\varphi) \equiv O\neg\varphi \\ \neg\mu X.\varphi \equiv \nu X.\neg\varphi(\neg X/X) \\ \hline \end{array}$$

表达能力： ν TL公式 $\nu X.(p \wedge OO(X))$ 不能用PLTL公式表示。

命题 5.13 ν TL的表达能力强于PLTL。

可满足性判定问题： 判定 ν TL公式是否可满足的问题称为 ν TL可满足性问题。 ν TL可满足性问题的复杂性为EXPTIME完全。

定义 5.7 给定标号Kripke模型 $M = \langle S, R, I, L \rangle$ 。 M 满足 ν TL公式 φ ，记作 $M \models \varphi$ ，当且仅当对所有 $\zeta \in [[M]]$ 有 $\langle S, \zeta, L \rangle \models \varphi$ 。

模型检测问题： 判定标号Kripke模型是否满足 ν TL公式的问题称为 ν TL模型检测问题。 ν TL模型检测问题的复杂性为PSPACE完全。

§5.2.2 PLTL公式到 ν TL公式的转换：

PLTL公式可以看成是 ν TL的一个子类，可用以下规则将PLTL公式转换到 ν TL公式。

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

命题 5.14 给定线性结构 $\langle S, \zeta, L \rangle$ 和PLTL公式 φ 。 $\langle S, \zeta, L \rangle \models \varphi$ 当且仅当 $\langle S, \zeta, L \rangle \models T(\varphi)$ 。

§5.2.3 公平标号Kripke结构上的LTL语义

符号集合AP上的LTL公式可在AP上的公平标号Kripke结构上解释。给定公平标号Kripke结构 $M = \langle S, R, I, L, F \rangle$ 且 $F = \{f_1, \dots, f_k\}$ 。

语义 设 $M' = \langle S, R, I, L \rangle$ 为 M 去掉公平约束的标号Kripke结构。 M 满足LTL公式 φ 当且仅当对所有 M 的公平计算 π ， $M', \pi \models \varphi$ 。我们有以下结论。

命题 5.15 $M \models \varphi$ 当且仅当 $M' \models (\bigwedge_{i=1}^k \square \diamond f_i) \rightarrow \varphi$ 。

§5.3 一阶线性时序逻辑

一阶线性时序逻辑用一阶逻辑来描述状态性质，具有表达能力强的优点。给定 $B = (F, P)$ 上的一阶逻辑。用 p 表示 WFF_B 中的任意公式。 B 上的一阶线性时序逻辑公式的集合 LTL 由以下语法给出。

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid \phi \leftrightarrow \phi \mid O\phi \mid \diamond\phi \mid \square\phi \mid \phi U \phi \mid \phi R \phi$$

给定 B 的解释 I 。设变量赋值的集合为 Σ 。

模型： LTL 公式的模型 M 为 Σ^ω 的子集。

语义： 设 $\zeta = \zeta_0\zeta_1\zeta_2 \cdots \in \Sigma^\omega$ 。 ζ 满足 φ ，记作 $\zeta \models \varphi$ ，定义如下。

$\zeta \models p$	若 $p \in WFF_B$ 且 $\zeta_0 \models_I p$
$\zeta \models \neg\varphi$	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 $\zeta \models \psi$
$\zeta \models \varphi \rightarrow \psi$	若 $\zeta \models \varphi$ 则 $\zeta \models \psi$
$\zeta \models \varphi \leftrightarrow \psi$	若 $\zeta \models \varphi \rightarrow \psi$ 且 $\zeta \models \psi \rightarrow \varphi$
$\zeta \models O\varphi$	若 $\zeta^1 \models \varphi$
$\zeta \models \square\psi$	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 $\exists i \geq 0. (\zeta^i \models \psi \text{ 且 } \forall 0 \leq j < i, \zeta^j \models \varphi)$
$\zeta \models \varphi R \psi$	若 $\forall i \geq 0. (\zeta^i \models \psi \text{ 或 } \exists 0 \leq j < i, \zeta^j \models \varphi)$

定义 5.8 $M \models \varphi$ ，当且仅当对所有 $\zeta \in M$ 都有 $\zeta \models \varphi$ 。

可满足性： 设 φ 为 LTL 公式。 φ 是可满足的，当且仅当存在 M 使得 $M \models \varphi$ 。

重言式和等价： 设 φ 和 ψ 为 LTL 公式。 φ 是重言式，记作 $\models \varphi$ ，当且仅当对任意 M 有 $M \models \varphi$ 。 φ 等价于 ψ ，记作 $\varphi \equiv \psi$ ，当且仅当 $\models \varphi \leftrightarrow \psi$ 。

完全集： 一阶线性时序逻辑中时序算子相互之间的可表达性和命题线性时序逻辑类似。我们有以下结论。

命题 5.16 $\{O, U\}$ 是 LTL 的时序算子极小完全集。

推理规则： 用 $\varphi \Rightarrow \psi$ 表示 $\square(\varphi \rightarrow \psi)$ 。设 $w, u \in QFF_B$ 为一元谓词公式(记其变量为 x)且 \sqsubseteq 为 P 中二元谓词符号。设 $W = \{\sigma(x) \mid I(w)(\sigma) = true\}$ 且 $U = \{\sigma(x) \mid I(w \wedge u)(\sigma) = true\}$ 。若 $(W, I_0(\sqsubseteq))$ 为 U 弱基集合，则称 (w, u, \sqsubseteq) 定义一个弱基集合。若 $(W, I_0(\sqsubseteq))$ 为良基集合，则称 (w, \sqsubseteq) 定义一良基集合。

设 $e \in T_B$ 为项, v 为没有在公式中出现过的变量, \sqsubseteq 为 P 中二元谓词符号, $w, u \in QFF_B$ 为一元谓词公式且 (w, u, \sqsubseteq) 定义一弱基集合。根据 LTL 公式的语义, 我们有以下推理规则。

$$\begin{array}{c}
 \frac{\varphi \Rightarrow \psi}{\varphi \Rightarrow \varphi_0 U \psi} \\
 \\
 \frac{\varphi \Rightarrow O(\psi)}{\varphi \Rightarrow O(\varphi_0 U \psi)} \\
 \\
 \begin{array}{l}
 \phi_2 \quad \Rightarrow \quad \psi_2 \\
 \phi_2 \wedge \neg \psi_1 \Rightarrow O(\psi_2 U \phi_2) \\
 \phi_1 \quad \Rightarrow \quad \psi_0 \wedge w_x^e \\
 \phi_1 \wedge \neg \psi_2 \Rightarrow \neg u_x^e \\
 \phi_1 \wedge e = v \Rightarrow O(\phi_1 U (\phi_2 \vee (\phi_1 \wedge e \sqsubseteq v))) \\
 \hline
 \varphi \quad \Rightarrow \quad (\phi_1 \vee \phi_2) \\
 \hline
 \varphi \quad \Rightarrow \quad \psi_0 U (\psi_1 R \psi_2)
 \end{array}
 \end{array}$$

由以上推理规则, 可得出如下规则用于形式为 $\psi_1 R \psi_2$ 和 $\psi_1 U \psi_2$ 的公式的推理 (其中 (w, \sqsubseteq) 定义一良基集合)。

$$\begin{array}{c}
 \begin{array}{l}
 \phi_2 \quad \Rightarrow \quad \psi_2 \\
 \phi_2 \wedge \neg \psi_1 \Rightarrow O(\psi_2 U \phi_2) \\
 \hline
 \varphi \quad \Rightarrow \quad \phi_2 \\
 \hline
 \varphi \quad \Rightarrow \quad \psi_1 R \psi_2
 \end{array}
 \qquad
 \begin{array}{l}
 \phi_1 \quad \Rightarrow \quad \psi_0 \wedge w_x^e \\
 \phi_1 \wedge e = v \Rightarrow O(\phi_1 U (\psi_1 \vee (\phi_1 \wedge e \sqsubseteq v))) \\
 \hline
 \varphi \quad \Rightarrow \quad (\phi_1 \vee \psi_1) \\
 \hline
 \varphi \quad \Rightarrow \quad \psi_0 U \psi_1
 \end{array}
 \end{array}$$

通过进一步化简可得以下推理规则(右边的规则可由左边规则得到)。

$$\begin{array}{c}
 \begin{array}{l}
 \varphi \Rightarrow \psi' \\
 \psi' \wedge \neg \varphi_0 \Rightarrow O\psi' \\
 \psi' \Rightarrow \psi \\
 \hline
 \varphi \Rightarrow \varphi_0 R \psi
 \end{array}
 \qquad
 \begin{array}{l}
 \varphi \Rightarrow \psi' \\
 \psi' \Rightarrow O\psi' \\
 \psi' \Rightarrow \psi \\
 \hline
 \varphi \Rightarrow \Box \psi
 \end{array} \\
 \\
 \begin{array}{l}
 \varphi \Rightarrow (\psi \vee \phi) \\
 \phi \Rightarrow \varphi_0 \wedge w_x^e \\
 \phi \wedge e = v \Rightarrow O(\psi \vee (\phi \wedge e \sqsubseteq v)) \\
 \hline
 \varphi \Rightarrow \varphi_0 U \psi
 \end{array}
 \qquad
 \begin{array}{l}
 \varphi \Rightarrow (\psi \vee \phi) \\
 \phi \Rightarrow w_x^e \\
 \phi \wedge e = v \Rightarrow O(\psi \vee (\phi \wedge e \sqsubseteq v)) \\
 \hline
 \varphi \Rightarrow \Diamond \psi
 \end{array}
 \end{array}$$

注：基于谓词逻辑的迁移模型通常对应于满足一定约束的 Σ^ω 的子集。比如我们可以对模型 M 添加以下约束。

$$\begin{array}{l} \text{若 } \zeta, \zeta' \in M \text{ 且 } \zeta_i = \zeta'_i, \quad \text{则 } \zeta_0 \cdots \zeta_i \zeta'_{i+1} \zeta'_{i+2} \cdots \in M \\ \text{若 } \zeta \in M \text{ 且 } \zeta_i = \zeta_j \text{ 且 } i < j, \quad \text{则 } \zeta_0 \cdots \zeta_{i-1} (\zeta_i \cdots \zeta_{j-1})^\omega \in M \\ \text{若 } \zeta \in M \text{ 且 } \zeta_i = \zeta_{i+1}, \quad \text{则 } \zeta = \zeta_0 \cdots \zeta_{i-1} (\zeta_i)^\omega \end{array}$$

§5.4 例子

例 5.1 (1) 用语义证明 $Op \equiv \neg O\neg p$ 。

证：给定任意标号 Kripke 模型 M 。设 ζ 为 M 的任意计算。我们有如下等价关系。

$$\begin{array}{l} \zeta \models Op \quad \Leftrightarrow \\ \zeta^1 \models p \quad \Leftrightarrow \\ \zeta^1 \not\models \neg p \quad \Leftrightarrow \\ \zeta \not\models O\neg p \quad \Leftrightarrow \\ \zeta \models \neg O\neg p \end{array}$$

因而 $Op \equiv \neg O\neg p$ 。

(2) 用语义证明 $pRq \equiv (qU(p \wedge q)) \vee \Box q$ 。

证：给定任意标号 Kripke 模型 M 。设 ζ 为 M 的任意计算。

(a) 从左到右的蕴含。

假设 $\zeta \models pRq$ 。

对任意 $i \geq 0$ ，若对所有 $j < i$ 都有 $\zeta^j \not\models p$ ，则 $\zeta^i \models q$ 。

若对所有 $j \geq 0$ 都有 $\zeta^j \not\models p$ ，那么对所有 $i \geq 0$ 有 $\zeta^i \models q$ ，因而 $\zeta \models \Box q$ 成立。

否则存在最小的 j 使得 $\zeta^j \models p$ ，那么对所有 $i \leq j$ 有 $\zeta^i \models q$ ，因而 $\zeta \models qU(p \wedge q)$ 成立。

(b) 从右到左的蕴含。

假设 $\zeta \models (qU(p \wedge q)) \vee \Box q$ 。

若 $\zeta \models \Box q$ ，那么对所有 $i \geq 0$ 有 $\zeta^i \models q$ ，因而 $\zeta \models pRq$ 成立。

否则 $\zeta \models (qU(p \wedge q))$ 成立，存在最小的 j 使得 $\zeta^j \models p \wedge q$ 且对所有 $i < j$ 有 $\zeta^i \models q$ ，因而 $\zeta \models (qU(p \wedge q))$ 成立。

(3) 用语义证明 $pRq \rightarrow \Box q$ 不成立。

证：构造 $AP = \{p, q\}$ 上的模型 $M = (S, R, I, L)$ 其中 $S = \{s_0, s_1\}$, $R = \{(s_0, s_1), (s_1, s_1)\}$, $I = \{s_0\}$, $L(s_0) = \{p, q\}$, $L(s_1) = \{\}$ 。

我们有 $M \models pRq$ 且 $M \not\models \Box q$ 。因而存在不满足 $pRq \rightarrow \Box q$ 的模型。

例 5.2 (1) O 算子不能用仅用 U 算子和命题逻辑连接符表示。

证：定义 $M = \langle S, R, I, L \rangle$ 其中 $S = \{s_0, s_1, s_2\}$, $R = \{(s_0, s_1), (s_1, s_2), (s_2, s_0)\}$, $I = \{s_0\}$, $L(s_0) = \{p\}$ 且 $L(s_2) = \{p\}$ 。

定义 $M' = \langle S, R, I', L \rangle$ 其中 $I' = \{s_1\}$ 。

我们有 $M \not\models Op$ 且 $M' \models Op$ 。

设 φ 为仅用 U 算子和命题逻辑连接符构造的公式。

若 φ 的长度为 l 。则 $M \models \varphi$ 当且仅当 $M' \models \varphi$ 。

假设对 φ_0, φ_1 有 $M \models \varphi_0$ 当且仅当 $M' \models \varphi_0$ 且 $M \models \varphi_1$ 当且仅当 $M' \models \varphi_1$ 。

设 φ 是用 φ_0, φ_1 和命题逻辑连接符构造的公式，则易知 $M \models \varphi$ 当且仅当 $M' \models \varphi$ 。

设 $\varphi = \varphi_0 U \varphi_1$ 。

设 $\zeta = s_0 s_1 s_2 (s_0 s_1 s_2)^\omega$ 。

分以下三种情况讨论。

- $\zeta^0 \models \varphi_1$ 。

由假设知 $\zeta^1 \models \varphi_1$ 。因而我们有 $\zeta^0 \models \varphi$ 且 $\zeta^1 \models \varphi$ 。

- $\zeta^0 \models \neg \varphi_1$ 且 $\zeta^0 \models \varphi_0$ 。

我们有 $\zeta^0 \models \varphi$ 当且仅当 $\zeta^1 \models \varphi$ 。

- $\zeta^0 \models \neg \varphi_1$ 且 $\zeta^0 \models \neg \varphi_0$ 。

由假设知 $\zeta^1 \models \neg \varphi_1$ 且 $\zeta^1 \models \neg \varphi_0$ 。因而我们有 $\zeta^0 \models \neg \varphi$ 且 $\zeta^1 \models \neg \varphi$ 。

因而 $M \models \varphi$ 当且仅当 $M' \models \varphi$ 。

由归纳法知对任意长度的仅用 U 算子和命题逻辑连接符的公式 φ ，我们有 $M \models \varphi$ 当且仅当 $M' \models \varphi$ 。

因而 Op 不等价于任何这样的公式。

(2) U 算子不能用仅用 O 算子和命题逻辑连接符表示。

证：定义 $M_k = \langle S_k, R, I, L \rangle$ 和 $M'_k = \langle S_k, R, I, L' \rangle$ 如下。

$$\begin{aligned} S_k &= \{s_0, s_1, \dots, s_k\} \\ R &= \{(s_0, s_1), (s_1, s_2), \dots, (s_{k-1}, s_k), (s_k, s_k)\} \\ I &= \{s_0\} \\ L(s_0) &= \dots = L(s_{k-1}) = L(s_k) = \{\} \\ L'(s_0) &= \dots = L'(s_{k-1}) = \{\} \\ L'(s_k) &= \{p\} \end{aligned}$$

我们有 $M_k \not\models (\neg p)Up$ 且 $M' \models (\neg p)Up$ 。

设 φ 为仅用 O 算子和命题逻辑连接符构造的公式。

若 φ 中 O 的个数为 l 。则对所有 $k \geq 1$ 有 $M_k \models \varphi$ 当且仅当 $M'_k \models \varphi$ 。

假设对 φ_0, φ_1 有对所有 $k \geq m$ 有 $M_k \models \varphi_0$ 当且仅当 $M'_k \models \varphi_0$ 且 $M_k \models \varphi_1$ 当且仅当 $M'_k \models \varphi_1$ 。

设 φ 是用 φ_0, φ_1 和命题逻辑连接符构造的公式，则易知对所有 $k \geq m$ 有 $M_k \models \varphi$ 当且仅当 $M'_k \models \varphi$ 。

设 $\varphi = O\varphi_0$ 。则易知对所有 $k \geq m+1$ 有 $M_k \models \varphi$ 当且仅当 $M'_k \models \varphi$ 。

由归纳法知对仅用 m 个 O 和任意多个命题逻辑连接符的公式 φ ，我们有 $M_{m+1} \models \varphi$ 当且仅当 $M'_{m+1} \models \varphi$ 。

因而 $(\neg p)Up$ 不等价于任何这样的一个用有限个 O 和任意多个命题逻辑连接符构造的公式。

例 5.3 (1) 用推理系统证明 $p \wedge \Box Op \rightarrow \Box p$ 。

证: 用 A_i 表示第 i 条时序逻辑公理。简单起见用 a, b, c 分别代表 $\Box Op, \Box(p \rightarrow Op), p \rightarrow \Box p$ 。我们证明 $p \wedge \Box Op \rightarrow \Box p$ 如下。

1	$Op \rightarrow (p \rightarrow Op)$	AX
2	$\Box(Op \rightarrow (p \rightarrow Op))$	$1, G$
3	$\Box Op \rightarrow \Box(p \rightarrow Op)$	$2, A_2, MP$
4	$(a \rightarrow b) \rightarrow ((b \rightarrow c) \rightarrow (a \rightarrow c))$	AX
5	$(b \rightarrow c) \rightarrow (a \rightarrow c)$	$4, 3, MP$
6	$\Box Op \rightarrow (p \rightarrow \Box p)$	$5, A_6, MP$
7	$(\Box Op \rightarrow (p \rightarrow \Box p)) \rightarrow (p \wedge \Box Op \rightarrow \Box p)$	AX
8	$p \wedge \Box Op \rightarrow \Box p$	$7, 6, MP$

若将 p 和 Op 列为前提条件, 则 $\Box p$ 的证明如下。

1	p	-
2	$\Box Op$	-
3	$Op \rightarrow (p \rightarrow Op)$	AX
4	$\Box(Op \rightarrow (p \rightarrow Op))$	$3, G$
5	$\Box Op \rightarrow \Box(p \rightarrow Op)$	$4, A_2, MP$
6	$\Box(p \rightarrow Op)$	$5, 2, MP$
7	$p \rightarrow \Box p$	$6, A_6, MP$
8	$\Box p$	$7, 1, MP$

(2) 用推理系统证明 $(Op \rightarrow Oq) \rightarrow O(p \rightarrow q)$ 。

证: 用 e, f 分别代表 $(Op \rightarrow Oq), O(p \rightarrow q)$ 。我们证明 $e \rightarrow f$ 如下。

1	$\neg p \rightarrow (p \rightarrow q)$	AX
2	$\Box(\neg p \rightarrow (p \rightarrow q))$	G
3	$O(\neg p \rightarrow (p \rightarrow q))$	A_4, MP
4	$O\neg p \rightarrow f$	A_8, MP
5	$(\neg Op \leftrightarrow O\neg p) \rightarrow ((O\neg p \rightarrow f) \rightarrow (\neg Op \rightarrow f))$	AX
6	$\neg Op \rightarrow O(p \rightarrow q)$	$A_7, 4, 5, MP$
7	$q \rightarrow (p \rightarrow q)$	AX
8	$\Box(q \rightarrow (p \rightarrow q))$	G
9	$O(q \rightarrow (p \rightarrow q))$	A_4, MP
10	$Oq \rightarrow f$	A_8, MP
11	$(\neg Op \rightarrow f) \rightarrow (Oq \rightarrow f) \rightarrow (e \rightarrow f)$	AX
12	$(e \rightarrow f)$	$6, 10, 11, MP$

例 5.4 设 r, g, a 为命题, φ 为以下公式的合取。

$$\begin{aligned} & \Box(r \wedge \neg a \rightarrow ((r \wedge \neg a)U(r \wedge a))) \\ & \Box(r \wedge a \rightarrow ((r \wedge a)U(g \wedge \neg a))) \\ & \Box(g \wedge \neg a) \rightarrow ((g \wedge \neg a)U((r \wedge \neg a) \vee (g \wedge a))) \\ & \Box(g \wedge a) \rightarrow ((g \wedge a)U(r \wedge \neg a)) \\ & \Box(\neg(r \wedge g)) \\ & r \wedge \neg a \end{aligned}$$

若 r, g 表示过街用的红绿灯的状态, a 表示行人按过过街绿色按钮 (重复按不产生效果) 的状态, $\neg a$ 表示行人可按过街绿色按钮的状态。则公式 φ 可表示以下描述:

- 初始状态为红灯且行人可按过街绿色按钮。
- 行人按了绿色按钮之后一段时间红灯变为绿灯, 且在这段时间内没法再按绿色按钮 (或者说重复按不产生效果)。
- 红灯变为绿灯之后, 可有效地重复按过街绿色按钮一次以保持绿灯。
- 若红灯变为绿灯之后没有行人按过街绿色按钮, 则一段时间之后绿灯变为红灯、返回初始状态。
- 若红灯变为绿灯之后有行人按过街绿色按钮, 则继续保持绿灯且在这段时间内没法再按绿色按钮。
- 若红灯变为绿灯之后且有行人按过过街绿色按钮, 则过一段时间之后绿灯变为红灯、返回初始状态。
- 红灯和绿灯不能同时出现。

例 5.5 (1) 设 AP 和 $K_3 = (S, R, I, L)$ 如例 2.3 中所定义。用限界语义证明 $K_3 \not\models \Diamond(p_3 \vee p_6)$ 。

证: 尝试找到 k 和从初始状态出发的 k 路径 π 使得 $K_3, \pi \models_k \Box \neg(p_3 \vee p_6)$ 成立。

(a) $k = 0$ 时有如下长度为 1 的初始状态出发的 k 路径: s_0 和 s_1 。这些路径均不满足所需条件。

(b) $k = 1$ 时有如下初始状态出发的 k 路径: $s_0s_{12}, s_1s_{12}, s_0s_{27}, s_1s_{27}$ 。这些路径均不满足所需条件。

(c) 检查 $k = 2$ 时所有初始状态出发的 k 路径, 其中 $\pi = s_0s_{27}s_{38}$ 是 (2, 1) 环路径且 $K_3, \pi \models_k \Box \neg(p_3 \vee p_6)$ 成立。因而 $K_3 \not\models \Diamond(p_3 \vee p_6)$ 。

(d) 由以上分析知 $(K_3, \Diamond(p_3 \vee p_6))$ 的最小完备阈值为 $k = 2$, 即用限界语义能够证明 $K_3 \not\models \Diamond(p_3 \vee p_6)$ 的最小 k 。

(2) 设 AP 和 S, R', I, L 如例 2.3 中所定义。令 $K'_3 = (S, R', I, L)$ 。用限界语义证明 $K'_3 \models \Diamond(p_3 \vee p_6)$ 。

证: 尝试寻找 k 和从初始状态出发的 k 路径 π 使得 $M, \pi \models_k \Box \neg(p_3 \vee p_6)$ 成立。

由限界语义理论知 $K'_3 \models \Diamond(p_3 \vee p_6)$ 的完备阈值不超过 $|M| \cdot 2^4$ 。

在 k 从零到完备阈值的范围不存在初始状态出发的 k 路径 π 使得 $K'_3, \pi \models_k \Box \neg(p_3 \vee p_6)$ 成立, 因而 $K'_3 \models \Diamond(p_3 \vee p_6)$ 成立。

例 5.6 (1) 将PLTL公式 Gp 化为等价的自动机的构造如下。

- (a) 将 Gp 写成 $\perp Rp$, 我们有初始节点 $n_0 = (\{\perp Rp\}, \{\}, \{\})$ 且 $a(n_0) = \{\epsilon\}$ 。
(b) 演化如下。

label	n	$a(n)$	eq
1	$\{\perp Rp\}, \{\}, \{\}$	$\{\epsilon\}$	11,12
11	$\{\perp, p\}, \{\perp Rp\}, \{\}$	$\{\epsilon\}$	11'
12	$\{p\}, \{\perp Rp\}, \{\perp Rp\}$	$\{\epsilon\}$	12'
11'	$\{\}, \{\perp, p, \perp Rp\}, \{\}$	$\{\epsilon\}$	0
12'	$\{\}, \{p, \perp Rp\}, \{\perp Rp\}$	$\{\epsilon\}$	
2	$\{\perp Rp\}, \{\}, \{\}$	$\{12'\}$	1

开始时我们有节点1, 然后转化为11和12, 然后这两个节点分别转化为11'和12'。节点11'由于不可满足被删除(最右列的值为0表示删除), 12'产生后继2, 2等同于1。

(c) 去掉完成转化的剩余的节点并且合并相同的节点, 最后剩下的节点的标号为12'。

(d) 我们有 $S = \{12'\}$, $I = \{12'\}$ 。由于不存在最外层时序算子为 U 的子公式, $F = \{\}$ 。由自动机的构造方法知 Σ 为 $\{p\}$ 的幂集。 $\Delta = \{(12', \{p\}, 12')\}$ 。

由以上构造知 $(\Sigma, S, \Delta, I, F)$ 是等价于 Gp 的泛Büchi自动机。

(2) 将PLTL公式 qUr 化为等价的自动机的构造如下。

- (a) 我们有初始节点 $n_0 = (\{qUr\}, \{\}, \{\})$ 且 $a(n_0) = \{\epsilon\}$ 。
(b) 演化如下。

label	n	$a(n)$	eq
1	$\{qUr\}, \{\}, \{\}$	$\{\epsilon\}$	11,12
11	$\{r\}, \{qUr\}, \{\}$	$\{\epsilon\}$	11'
12	$\{q\}, \{qUr\}, \{qUr\}$	$\{\epsilon\}$	12'
11'	$\{\}, \{qUr, r\}, \{\}$	$\{\epsilon\}$	
12'	$\{\}, \{qUr, q\}, \{qUr\}$	$\{\epsilon\}$	
2	$\{\}, \{\}, \{\}$	$\{\epsilon, 11'\}$	
3	$\{qUr\}, \{\}, \{\}$	$\{\epsilon, 12'\}$	1
4	$\{\}, \{\}, \{\}$	$\{\epsilon, 2\}$	2

开始时我们有节点1, 然后转化为11和12, 然后这两个节点分别转化为11'和12'。由11'产生后继2, 由12'产生后继3。由2产生后继4。3和4分别等同于2和1。

(c) 去掉完成转化的节点并且合并相同的节点, 最后剩下的节点的标号为11', 12', 2。

(d) 我们有 $S = \{11', 12', 2\}$, $I = \{11', 12'\}$, $F = \{\{11', 2\}\}$ 。由自动机的构造方法知 Σ 为 $\{q, r\}$ 的幂集。用 a 表示 $\{q, r\}$ 的子集。 Δ 为以下集合的并集。

$$\begin{aligned} & \{(11', a, 2) \mid a \models r\} \\ & \{(12', a, 31') \mid a \models q\} \\ & \{(2, a, 2) \mid a \subseteq \{q, r\}\} \end{aligned}$$

由以上构造知 $(\Sigma, S, \Delta, I, F)$ 是等价于 qUr 的泛Büchi 自动机。

(3) 将PLTL公式 $pU(qUr)$ 化为等价的自动机的构造如下。

(a) 我们有初始节点 $n_0 = (\{pU(qUr)\}, \{\}, \{\})$ 且 $a(n_0) = \{\epsilon\}$ 。

(b) 演化如下。

label	n	$a(n)$	eq
1	$\{pU(qUr)\}, \{\}, \{\}$	$\{\epsilon\}$	11,12
11	$\{qUr\}, \{pU(qUr)\}, \{\}$	$\{\epsilon\}$	111,112
12	$\{p\}, \{pU(qUr)\}, \{pU(qUr)\}$	$\{\epsilon\}$	12'
111	$\{r\}, \{pU(qUr), qUr\}, \{\}$	$\{\epsilon\}$	111'
112	$\{q\}, \{pU(qUr), qUr\}, \{, qUr\}$	$\{\epsilon\}$	112'
111'	$\{\}, \{pU(qUr), qUr, r\}, \{\}$	$\{\epsilon\}$	
112'	$\{\}, \{pU(qUr), qUr, q\}, \{qUr\}$	$\{\epsilon\}$	
12'	$\{\}, \{pU(qUr), p\}, \{pU(qUr)\}$	$\{\epsilon\}$	
2	$\{\}, \{\}, \{\}$	$\{\epsilon, 111'\}$	
3	$\{qUr\}, \{\}, \{\}$	$\{\epsilon, 112'\}$	31,32
4	$\{pU(qUr)\}, \{\}, \{\}$	$\{\epsilon, 12'\}$	1
31	$\{r\}, \{qUr\}, \{\}$	$\{\epsilon, 112'\}$	31'
32	$\{q\}, \{qUr\}, \{qUr\}$	$\{\epsilon, 112'\}$	32'
31'	$\{\}, \{qUr, r\}, \{\}$	$\{\epsilon, 112'\}$	
32'	$\{\}, \{qUr, q\}, \{qUr\}$	$\{\epsilon, 112'\}$	
5	$\{\}, \{\}, \{\}$	$\{\epsilon, 111', 2\}$	2
6	$\{\}, \{\}, \{\}$	$\{\epsilon, 112', 31'\}$	2
7	$\{qUr\}, \{\}, \{\}$	$\{\epsilon, 112', 32'\}$	3

(c) 去掉完成转化的节点并且合并相同的节点, 最后剩下的节点的标号为111', 112', 12', 2, 31', 32'。

(d) 我们有

$$S = \{111', 112', 12', 2, 31', 32'\},$$

$$I = \{111', 112', 12'\},$$

$$F = \{\{111', 112', 2, 31', 32'\}, \{111', 12', 2, 31'\}\}.$$

由自动机的构造方法知 Σ 为 $\{p, q, r\}$ 的幂集。用 a 表示 $\{p, q, r\}$ 的子集。 Δ 为以下集合的并集。

$$\begin{aligned} & \{(111', a, 2) \mid a \models r\} & \{(112', a, 31') \mid a \models q\} \\ & \{(112', a, 32') \mid a \models q\} & \{(12', a, 111') \mid a \models p\} \\ & \{(12', a, 112') \mid a \models p\} & \{(12', a, 12') \mid a \models p\} \\ & \{(2, a, 2) \mid a \subseteq \{p, q, r\}\} & \{(31', a, 2) \mid a \models r\} \\ & \{(32', a, 2) \mid a \models q\} & \{(32', a, 32') \mid a \models q\} \end{aligned}$$

由以上构造知 $(\Sigma, S, \Delta, I, F)$ 是等价于 $pU(qUr)$ 的泛Büchi 自动机。

§5.5 练习

1. 用PLTL语义证明 $\Box(p \rightarrow Op) \rightarrow (p \rightarrow \Box p)$ 成立且解释这个蕴含关系反过来不成立。
2. 用PLTL推理系统证明 $O(p \vee q) \leftrightarrow (Op \vee Oq)$ 。
3. 用PLTL写下信号灯变化的规范：信号灯依次序绿黄红变化，每个状态有且只有一个信号，初始信号为绿色，黄色只停留一个状态，红绿色可以连续在多个状态上成立。
4. 设 AP 和 $K'_3 = (S, R', I, L)$ 如例5.5中所定义。设 $\varphi_1 = (p_1 \vee p_2)Up_3$ 和 $\varphi_2 = (p_1 \vee p_2)U(p_3 \vee p_6)$ 。用限界语义证明 $K'_3 \models \varphi_1$ 不成立且 $K'_3 \models \varphi_2$ 成立。
5. 分别构造与公式 $(p \vee (qUr))$ 和 $(Xp \wedge (qRr))$ 等价的自动机。

§6 分枝时序逻辑

本章介绍可用以描述程序模型性质的分枝时序逻辑。抽象模型中的一个状态通常有多个可能的后继状态和多条可能从该状态出发的路径。线性时序逻辑仅表达线性运行过程的性质。表达这类分枝路径的性质则需要逻辑中添加路径量词。

§6.1 计算树逻辑CTL

考虑建立在命题逻辑上的计算树逻辑，记作CTL。给定一个原子命题集合 AP 。用 p 表示 AP 中的任意命题。PLTL公式的集合由以下语法给出。

$$\begin{aligned} \phi ::= & p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \phi \rightarrow \phi \mid \phi \leftrightarrow \phi \mid \\ & EX\phi \mid EF\phi \mid EG\phi \mid E(\phi U \phi) \mid E(\phi R \phi) \mid AX\phi \mid AF\phi \mid AG\phi \mid A(\phi U \phi) \mid A(\phi R \phi) \end{aligned}$$

符号 $EX, EF, EG, EU, ER, AX, AF, AG, AU, AR$ 称为CTL的时序算子。

模型： AP 上的CTL公式在 AP 上的标号Kripke结构上解释。

CTL公式的语义： 给定 $M = \langle S, R, I, L \rangle$ 。状态 $s \in S$ 满足CTL公式 φ ，记为 $M, s \models \varphi$ ，在 M 已确定的情况亦写做 $s \models \varphi$ ，定义如下。

$s \models p$	若 $p \in AP$ 且 $p \in L(s)$
$s \models \neg\varphi$	若 $s \not\models \varphi$
$s \models \varphi \vee \psi$	若 $s \models \varphi$ 或 $s \models \psi$
$s \models \varphi \wedge \psi$	若 $s \models \varphi$ 且 $s \models \psi$
$s \models \varphi \rightarrow \psi$	若 $s \models \varphi$ 则 $s \models \psi$
$s \models \varphi \leftrightarrow \psi$	若 $s \models \varphi \rightarrow \psi$ 且 $s \models \psi \rightarrow \varphi$
$s \models EX\varphi$	若存在 s 的后继 v 使得 $v \models \varphi$
$s \models EG\psi$	若存在 s 为起点的无穷路径 ζ 使得 $\forall i \geq 0, \zeta_i \models \psi$
$s \models EF\psi$	若存在 s 为起点的无穷路径 ζ 使得 $\exists i \geq 0, \zeta_i \models \psi$
$s \models E(\varphi U \psi)$	若存在 s 为起点的无穷路径 ζ 使得 $\exists i \geq 0. (\zeta_i \models \psi$ 且 $\forall 0 \leq j < i, \zeta_j \models \varphi)$
$s \models E(\varphi R \psi)$	若存在 s 为起点的无穷路径 ζ 使得 $\forall i \geq 0. (\zeta_i \models \psi$ 或 $\exists 0 \leq j < i, \zeta_j \models \varphi)$
$s \models AX\varphi$	若对所有 s 的后继 v 有 $v \models \varphi$
$s \models AG\psi$	若对所有 s 为起点的无穷路径 ζ 有 $\forall i \geq 0, \zeta_i \models \psi$
$s \models AF\psi$	若对所有 s 为起点的无穷路径 ζ 有 $\exists i \geq 0, \zeta_i \models \psi$
$s \models A(\varphi U \psi)$	若对所有 s 为起点的无穷路径 ζ 有 $\exists i \geq 0. (\zeta_i \models \psi$ 且 $\forall 0 \leq j < i, \zeta_j \models \varphi)$
$s \models A(\varphi R \psi)$	若对所有 s 为起点的无穷路径 ζ 有 $\forall i \geq 0. (\zeta_i \models \psi$ 或 $\exists 0 \leq j < i, \zeta_j \models \varphi)$

定义 6.1 $M \models \varphi$ 当且仅当对所有 $s \in I$ 有 $M, s \models \varphi$ 。

可满足性： 设 φ 为CTL公式。 φ 是可满足的，当且仅当存在标号Kripke结构 M 使得 $M \models \varphi$ 。

重言式和等价： 设 φ 和 ψ 为CTL公式。 φ 是重言式，记作 $\models \varphi$ ，当且仅当对标号Kripke结构 M 有 $M \models \varphi$ 。 φ 等价于 ψ ，记作 $\varphi \equiv \psi$ ，当且仅当 $\models \varphi \leftrightarrow \psi$ 。

等价公式： 设 φ 和 ψ 为CTL公式。我们有以下等价的公式对。

$AX\varphi$	\equiv	$\neg EX\neg\varphi$
$AG\varphi$	\equiv	$\neg EF\neg\varphi$
$AF\varphi$	\equiv	$\neg EG\neg\varphi$
$A(\varphi R\psi)$	\equiv	$\neg E(\neg\varphi U\neg\psi)$
$A(\varphi U\psi)$	\equiv	$\neg E(\neg\varphi R\neg\psi)$
$E(\varphi U\psi)$	\equiv	$\psi \vee (\varphi \wedge EXE(\psi U(\varphi \wedge \psi)))$
$E(\varphi R\psi)$	\equiv	$\psi \wedge (\varphi \vee EXE(\psi U(\varphi \wedge \psi)))$
$EF\varphi$	\equiv	$E(\top U\varphi)$
$EG\varphi$	\equiv	$E(\neg\top R\varphi)$
$E(\varphi R\psi)$	\equiv	$E(\psi U(\varphi \wedge \psi)) \vee EG\psi$

命题 6.1 $\{EX, EG, EU\}$ 是CTL的时序算子极小完全集。

NNF范式： 只使用逻辑连接符 \wedge, \vee, \neg 且逻辑联接符 \neg 只出现在命题前面的公式称为NNF范式。

命题 6.2 每个CTL公式等价于一个CTL的NNF范式。

对每个CTL公式，我们可以将其转换成等价的只使用 $\{\wedge, \vee, \neg\}$ 中的命题逻辑联接符和 $\{EX, EG, EU\}$ 中的时序算子的公式，然后再继续使用以上提到的等价关系和以下与命题逻辑联接符相关的等价关系将其转换成时序算子在 $\{EX, ER, EU, AX, AR, AU\}$ 中的NNF范式公式。

$\neg\neg\varphi$	\equiv	φ
$\neg(\varphi \vee \psi)$	\equiv	$\neg\varphi \wedge \neg\psi$
$\neg(\varphi \wedge \psi)$	\equiv	$\neg\varphi \vee \neg\psi$

可满足性判定问题： 判定CTL公式是否可满足的问题称为CTL可满足性问题。CTL可满足性问题的复杂性为EXPTIME完全。

模型检测问题： 判定模型是否满足CTL公式的问题称为CTL模型检测问题。CTL模型检测问题的复杂性为P完全。

§6.1.1 CTL公式的推理

以 $\{EX, EU, EG, EF, AX, AU, AG, AF\}$ 为CTL公式的时序算子集。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。设 p, q, r 为CTL公式。

时序逻辑公理：

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

命题逻辑推理系统：

(AX) 若 p 是命题逻辑的重言式的实例，则 p 是公理。

(MP) 若 $\vdash p \rightarrow q$ 且 $\vdash p$ ，则 $\vdash q$ 。

时序推理规则：

(G) 若 $\vdash p$ ，则 $\vdash AGp$ 。

命题 6.3 CTL推理系统是可靠且完备的。

§6.1.2 CTL限界语义

给定有穷状态标号Kripke结构 $M = \langle S, R, I, L \rangle$ 。

称长度 $k + 1$ 的路径为 k 路径。记 k 路径的集合为 P_k 。

以下考虑仅用 AX, AR, AU, EX, ER, EU 时序算子的NNF范式的CTL公式。

用 $\pi \in P_k(s)$ 表示 π 是以 s 为起点的 k 路径。

用 $\gamma(\pi)$ 表示 π 中有重复出现的状态，定义如下。

$$\gamma(\pi) = \bigvee_{i=0}^{k-1} \bigvee_{j=i+1}^k \pi_i = \pi_j$$

用 $\gamma_0(M, \pi, k, \varphi)$ 表示以下公式（用于定义6.2中限界语义 \models_k 的定义）。

$$\bigwedge_{i=0}^k (M, \pi_i \not\models_k \varphi) \rightarrow \gamma(\pi)$$

定义 6.2 设 φ 为CTL公式。限界语义 $M, s \models_k \varphi$ 定义如下。

$M, s \models_k p$	若 $p \in AP$ 且 $p \in L(s)$
$M, s \models_k \neg p$	若 $M, s \not\models_k p$
$M, s \models_k \varphi \vee \psi$	若 $M, s \models_k \varphi$ 或 $M, s \models_k \psi$
$M, s \models_k \varphi \wedge \psi$	若 $M, s \models_k \varphi$ 且 $M, s \models_k \psi$
$M, s \models_k AX\varphi$	若 $k \geq 1$ 且 $\forall \pi \in P_k(s)$, $M, \pi_1 \models_k \varphi$
$M, s \models_k A(\varphi U \psi)$	若 $\forall \pi \in P_k(s)$, $\exists i \leq k$, $M, \pi_i \models_k \psi$ 且 $\forall j < i$, $M, \pi_j \models_k \varphi$
$M, s \models_k A(\varphi R \psi)$	若 $\forall \pi \in P_k(s)$, $(\forall i \leq k. (\forall j < i. (M, \pi_j \not\models_k \varphi) \rightarrow (M, \pi_i \models_k \psi)))$ 且 $\gamma_0(M, \pi, k, \varphi)$
$M, s \models_k EX\varphi$	若 $k \geq 1$ 且 $\exists \pi \in P_k(s)$, $M, \pi_1 \models_k \varphi$
$M, s \models_k E(\varphi U \psi)$	若 $\exists \pi \in P_k(s)$, $\exists i \leq k$, $M, \pi_i \models_k \psi$ 且 $\forall j < i$, $M, \pi_j \models_k \varphi$
$M, s \models_k E(\varphi R \psi)$	若 $\exists \pi \in P_k(s)$, $(\forall i \leq k. (\forall j < i. (M, \pi_j \not\models_k \varphi) \rightarrow (M, \pi_i \models_k \psi)))$ 且 $\gamma_0(M, \pi, k, \varphi)$

命题 6.4 对所有 $k \geq 0$, 若 $M, s \models_k \varphi$, 则 $M, s \models_{k+1} \varphi$ 。

正确性与完备性:

命题 6.5 $M, s \models \varphi$ 当且仅当存在 $k \geq 0$ 使得 $M, s \models_k \varphi$ 。

用 $M \models_k \varphi$ 表示对所有初始状态 s 有 $M, s \models_k \varphi$ 。

推论 6.1 $M \models \varphi$ 当且仅当存在 $k \geq 0$ 使得 $M \models_k \varphi$ 。

推论 6.2 $M \not\models \varphi$ 当且仅当存在 $k \geq 0$ 和 $s \in I$ 使得 $M, s \models_k \neg\varphi$ 。

§6.2 CTL公式的不动点表示与模态 μ 演算

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:

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

我们有以下等式:

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

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 τ_1 和 τ_2 的定义知 τ_1 和 τ_2 有最小和最大不动点。 根据 CTL 语义知 $[[EGp]]$ 是 τ_1 的最大不动点, $[[E(pUq)]]$ 是 τ_2 的最小不动点, 即

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

类似地, 其它模态算子也可以用不动点刻画。

定义 $ax(A) = \{s \in S \mid \forall s' \in S. ((s, s') \in R \rightarrow s' \in A)\}$ 。 则 $[[AXp]] = ax([[p]])$ 。

为方便书写, 我们直接将 p, q 看成 S 的子集, 时序算子和命题逻辑联接符看成是 2^S 上的函数。

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

命题 6.6 给定 Kripke 结构 M 和 CTL 公式 φ 。 $M, s \models \varphi$ 当且仅当 $s \in [[\varphi]]$ 。

推论 6.3 给定标号 Kripke 结构 M 和 CTL 公式 φ 。 $M \models \varphi$ 当且仅当 $I \in [[\varphi]]$ 。

§6.2.1 模态 μ 演算

由关于CTL的讨论我们知道CTL公式可以用时序算子 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。称为模态 μ 演算或简称为 μ 演算。给定一个原子命题集合 AP ,一个变量集合 V 。用 p 表示 AP 中任意命题, X 表示 V 中任意变量。一种简单的 μ -演算的公式的集合可由以下语法给出。

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

其中 $\mu X. \phi$ 和 $\nu X. \phi$ 的 ϕ 中不受围的 X 必须在偶数个 \neg 符号的作用范围之下。

带动作的描述: 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 。用 p 表示 AP 中任意命题, X 表示 V 中任意变量, a 表示 A 中任意动作。 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 $\mu X. \phi$ 和 $\nu X. \phi$ 的 ϕ 中不受围的 X 必须在偶数个 \neg 符号的作用范围之下。

μ 演算公式在双标号迁移系统上的语义解释: 设 $M = \langle \Sigma, S, \Delta, I, L \rangle$ 为原子命题集 AP 上的双标号迁移系统, $e: V \rightarrow 2^S$ 为变量到 S 子集的赋值。 μ 演算公式的语义如下:

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

所有变量都是受围变量的公式称为闭公式。闭公式的语义不受 e 的影响。设 M 为双标号迁移系统, φ 为闭公式。 φ 在 M 中的语义可写为 $[[\varphi]]$ 。

定义 6.3 $M \models \varphi$ 当且仅当 $I \subseteq [[\varphi]]$ 。

可满足性: 设 φ 为 μ 演算公式。 φ 是可满足的,当且仅当存在 M 使得 $M \models \varphi$ 。

重言式和等价: 设 φ 为 μ 演算公式。 φ 是重言式,记作 $\models \varphi$,当且仅当对任意 M 有 $M \models \varphi$ 。 φ 等价于 ψ ,记作 $\varphi \equiv \psi$,当且仅当 $\models (\varphi \wedge \psi) \vee (\neg\varphi \wedge \neg\psi)$ 。

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。

$$\begin{array}{l} \hline [a]\phi \equiv \neg\langle a \rangle\neg\phi \\ \mu X.\phi \equiv \neg\nu X.\neg\phi(\neg X/X) \\ \hline \end{array}$$

表达能力： μ 演算公式 $\nu X.(p \wedge [.]X)$ 不能用CTL公式表示。

命题 6.7 μ 演算的表达能力强于CTL。

可满足性判定问题： 判定 μ 演算公式是否可满足的问题复杂性为EXPTIME完全。

模型检测问题： 判定双标号迁移系统是否满足 μ 演算公式的问题复杂性为 $NP \cap coNP$ 。

§6.2.2 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，可将CTL所描述系统中的动作都归为同一类。用点表示动作，则可用以下规则将CTL公式转换到 μ 演算公式。

$$\begin{array}{l} T(p) = p \\ T(\neg\phi) = \neg T(\phi) \\ T(\phi \vee \psi) = T(\phi) \vee T(\psi) \\ T(EX\phi) = \langle \cdot \rangle T(\phi) \\ T(E(\phi U \psi)) = \mu Y.(T(\psi) \vee (T(\phi) \wedge \langle \cdot \rangle Y)) \\ T(EG\phi) = \nu Y.(T(\phi) \wedge \langle \cdot \rangle Y) \end{array}$$

命题 6.8 给定AP上的标号Kripke模型 $M = \langle S, R, I, L \rangle$ 。定义双标号迁移系统 $M' = \langle \Sigma, S, \Delta, I, L \rangle$ 其中 $\Sigma = \{\cdot\}$ 和 $\Delta = \{(s, a, s') \mid (s, s') \in R, a \in \Sigma\}$ 。设 ϕ 为CTL公式。 $M \models \phi$ 当且仅当 $M' \models T(\phi)$ 。

§6.2.3 公平标号Kripke结构下的CTL语义

符号集合AP上的CTL公式可在AP上的公平标号Kripke结构上解释。给定公平标号Kripke结构 $M = \langle S, R, I, L, F \rangle$ 且 $F = \{f_1, \dots, f_k\}$ 。

语义 状态 $s \in S$ 满足CTL公式 ϕ ，记为 $M, s \models \phi$ ，其语义与CTL公式在标号Kripke结构上解释的解释类似，仅将存在 s 起点的无穷路径换成存在 s 起点的公平路径，对所有 s 起点的无穷路径换成对所有 s 起点的公平路径。

不动点表示 为方便区分，用 EX_f, EU_f, EG_f 表示公平标号Kripke结构上的CTL性质的相应算子，依然用 EX, EU, EG 表示去掉公平约束的标号Kripke结构上的CTL性质。用 $fair$ 表示公平状态集合。将公式理解为状态集合。我们有以下等式。

$$\begin{array}{l} \hline fair = \nu Z.(\bigwedge_{i=1}^k EX(EF(Z \wedge f_i))) \\ EX_f p = EX(p \wedge fair) \\ E(p U_f q) = E(p U (q \wedge fair)) \\ EG_f p = \nu Z.(p \wedge \bigwedge_{i=1}^k EX(E(p U (Z \wedge f_i)))) \\ \hline \end{array}$$

设 φ 为时序算子在 $\{EX_f, EU_f, EG_f\}$ 中的CTL公式。

命题 6.9 $M \models \varphi$ 当且仅当 $I \subseteq \varphi$ 。

§6.3 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。用 p 表示AP中任意命题。CTL*公式的集合由以下语法给出。

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

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

模型: AP上的CTL*公式在AP上的标号Kripke结构上解释。

CTL*语义: 设 $M = \langle S, R, I, L \rangle$ 是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 s 为起点的路径 $\pi: M, \pi \models \varphi$
$M, s \models E\varphi$	若存在 M 中 s 为起点的路径 $\pi: M, \pi \models \varphi$
$M, \pi \models p$	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 $\forall i \geq k, \forall 0 \leq j < i, M, \pi^j \not\models \varphi$ 则 $M, \pi^i \models \psi$

定义 6.4 设 φ 为CTL*状态公式, M 为Kripke结构。 $M \models \varphi$ 当且仅当对所有 $s \in I, M, s \models \varphi$ 。

可满足公式： 设 φ 为CTL*状态公式。 φ 是可满足的，当且仅当存在 M 使得 $M \models \varphi$ 。

重言式和等价： 设 φ 为CTL*状态公式。 φ 是重言式，记作 $\models \varphi$ ，当且仅当对任意 M 有 $M \models \varphi$ 。 φ 等价于 ψ ，记作 $\varphi \equiv \psi$ ，当且仅当 $\models (\varphi \wedge \psi) \vee (\neg\varphi \wedge \neg\psi)$ 。

表达能力： 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。我们有以下结论。

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。

命题 6.10 PLTL和CTL的表达能力存在互补关系且CTL*的表达能力强于PLTL和CTL的并集。

可满足性判定问题： 判定CTL*公式是否可满足的问题复杂性为双指数时间完全。

模型检测问题： 判定标号Kripke模型是否满足CTL*公式的问题复杂性为PSPACE完全。

§6.4 例子

例 6.1 设 $AP = \{p_0, p_1, p_2, p_3, p_4, q_0, q_1, q_2\}$ 。定义 S, R, I, L 如下。

$$S = \{s_0, s_1, s_2, s_3, s_4, s_5\}。$$

$$R = \{(s_0, s_1), (s_0, s_2), (s_1, s_3), (s_1, s_5), (s_2, s_5), (s_2, s_4), (s_3, s_4), (s_3, s_5), (s_4, s_5), (s_5, s_0)\}。$$

$$I = \{s_0\}。$$

L 定义如下。

$$L(s_0) = \{p_0, q_0\}$$

$$L(s_1) = \{p_1, q_0\}$$

$$L(s_2) = L(s_3) = \{p_2, q_0\}$$

$$L(s_4) = \{p_4, q_1\}$$

$$L(s_5) = \{p_3, q_2\}$$

则 $K = (S, R, I, L)$ 为标号Kripke模型。

(1) 对CTL性质 $\varphi_0 = E(q_0 U q_2)$ 和CTL性质 $\varphi_1 = AG(q_0 \vee q_2)$ ，我们有如下结论。

$$\begin{array}{cccc} K \not\models_0 \varphi_0 & K \not\models_0 \neg\varphi_0 & K \not\models_0 \varphi_1 & K \not\models_0 \neg\varphi_1 \\ K \not\models_1 \varphi_0 & K \not\models_1 \neg\varphi_0 & K \not\models_1 \varphi_1 & K \not\models_1 \neg\varphi_1 \\ K \models_2 \varphi_0 & K \not\models_2 \neg\varphi_0 & K \not\models_2 \varphi_1 & K \models_2 \neg\varphi_1 \end{array}$$

因而模型满足 φ_0 且可以确定模型满足 $E(q_0 U q_2)$ 的最小的界为 $k = 2$ ，模型不满足 φ_1 且可以确定模型不满足 $AG(q_0 \vee q_2)$ 的最小的界同样为 $k = 2$ 。

例 6.2 设 AP 和 $K = (S, R, I, L)$ 如例6.1中所定义。

(1) 我们有 $E(q_0 U q_2) = \mu Z.(q_2 \vee (q_0 \wedge EXZ))$ 。根据不动点算法, $[[E(q_0 U q_2)]]$ 的计算过程如下。

$$\begin{aligned}
S_0 &= false & &= \{\} \\
S_1 &= q_2 & &= \{s_5\} \\
S_2 &= \{s_5\} \cup (\{s_0, s_1, s_2, s_3\} \cap \{s_1, s_2, s_3, s_4\}) & &= \{s_1, s_2, s_3, s_5\} \\
S_3 &= \{s_5\} \cup (\{s_0, s_1, s_2, s_3\} \cap \{s_0, s_1, s_2, s_3, s_4\}) & &= \{s_0, s_1, s_2, s_3, s_5\} \\
S_4 &= \{s_5\} \cup (\{s_0, s_1, s_2, s_3\} \cap \{s_0, s_1, s_2, s_3, s_4, s_5\}) & &= S_3
\end{aligned}$$

因而 $[[E(q_0 U q_2)]] = \{s_0, s_1, s_2, s_3, s_5\}$ 。

(2) 我们有 $AG(q_0 \vee q_2) = \nu Z.((q_0 \vee q_2) \wedge AXZ)$ 。根据不动点算法, $[[AG(q_0 \vee q_2)]]$ 的计算过程如下。

$$\begin{aligned}
S_0 &= true & &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\
S_1 &= q_0 \vee q_2 & &= \{s_0, s_1, s_2, s_3, s_5\} \\
S_2 &= \{s_0, s_1, s_2, s_3, s_5\} \cap \{s_0, s_1, s_4, s_5\} & &= \{s_0, s_1, s_5\} \\
S_3 &= \{s_0, s_1, s_2, s_3, s_5\} \cap \{s_4, s_5\} & &= \{s_5\} \\
S_4 &= \{s_0, s_1, s_2, s_3, s_5\} \cap \{s_4\} & &= \{\} \\
S_5 &= \{s_0, s_1, s_2, s_3, s_5\} \cap \{\} & &= S_4
\end{aligned}$$

因而 $[[AG(q_0 \vee q_2)]] = \{\}$ 。

(3) 由以上计算知模型满足 $E(q_0 U q_2)$ 且模型不满足 $AG(q_0 \vee q_2)$ 。

例 6.3 设 AP 和 S, R, I, L 如例 6.1 中所定义。设 $\Sigma = \{1, 2, qc, qq\}$ 。定义 Δ 为以下集合的并集。

$$\begin{aligned}
&\{(s_0, 1, s_1), (s_0, 2, s_2)\} \\
&\{(s_1, 1, s_3), (s_1, 2, s_5)\} \\
&\{(s_2, 1, s_5), (s_2, 2, s_4)\} \\
&\{(s_3, 2, s_4), (s_3, 1, s_5)\} \\
&\{(s_4, qq, s_5), (s_5, qc, s_0)\}
\end{aligned}$$

则 $K' = (\Sigma, S, \Delta, I, L)$ 为双标号迁移系统。

(1) $\mu Z.(q_2 \vee \langle 1 \rangle Z)$ 的计算过程如下。

$$\begin{aligned}
S_0 &= false & &= \{\} \\
S_1 &= q_2 \vee \langle 1 \rangle \{\} & &= \{s_5\} \\
S_2 &= \{s_5\} \cup \{s_2, s_3\} & &= \{s_2, s_3, s_5\} \\
S_3 &= \{s_5\} \cup \{s_1, s_2, s_3\} & &= \{s_1, s_2, s_3, s_5\} \\
S_4 &= \{s_5\} \cup \{s_0, s_1, s_2, s_3\} & &= \{s_0, s_1, s_2, s_3, s_5\} \\
S_5 &= \{s_5\} \cup \{s_0, s_1, s_2, s_3\} & &= S_4
\end{aligned}$$

因而 $\mu Z.(q_2 \vee \langle 1 \rangle Z) = \{s_0, s_1, s_2, s_3, s_5\}$ 。

(2) $\nu Z.(\neg q_2 \wedge [1]Z)$ 的计算过程如下。

$$\begin{aligned}
S_0 &= true &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\
S_1 &= \neg q_2 \wedge [1]\{s_0, s_1, s_2, s_3, s_4, s_5\} &= \{s_0, s_1, s_2, s_3, s_4\} \\
S_2 &= \{s_0, s_1, s_2, s_3, s_4\} \cap \{s_0, s_1, s_4, s_5\} &= \{s_0, s_1, s_4\} \\
S_3 &= \{s_0, s_1, s_2, s_3, s_4\} \cap \{s_0, s_4, s_5\} &= \{s_0, s_4\} \\
S_4 &= \{s_0, s_1, s_2, s_3, s_4\} \cap \{s_4, s_5\} &= \{s_4\} \\
S_5 &= \{s_0, s_1, s_2, s_3, s_4\} \cap \{s_4, s_5\} &= S_4
\end{aligned}$$

因而 $\nu Z.(\neg q_2 \wedge [1]Z) = \{s_4\}$ 。

由 ν 和 μ 的对偶关系亦可知 $\nu Z.(\neg q_2 \wedge [1]Z) = \neg \mu Z.(q_2 \vee \langle 1 \rangle Z) = \{s_4\}$ 。

(3) $\mu Z.(\neg q_2 \vee [1]Z)$ 的计算过程如下。

$$\begin{aligned}
S_0 &= false &= \{\} \\
S_1 &= q_2 \vee [1]\{\} &= \{s_4, s_5\} \\
S_2 &= \{s_5\} \cup \{s_2, s_3, s_4, s_5\} &= \{s_2, s_3, s_4, s_5\} \\
S_3 &= \{s_5\} \cup \{s_1, s_2, s_3, s_4, s_5\} &= \{s_1, s_2, s_3, s_4, s_5\} \\
S_4 &= \{s_5\} \cup \{s_0, s_1, s_2, s_3, s_4, s_5\} &= \{s_0, s_1, s_2, s_3, s_4, s_5\} \\
S_5 &= \{s_5\} \cup \{s_0, s_1, s_2, s_3, s_4, s_5\} &= S_4
\end{aligned}$$

因而 $\mu Z.(\neg q_2 \vee [1]Z) = \{s_0, s_1, s_2, s_3, s_4, s_5\}$ 。

(4) 由以上计算知模型满足 $\mu Z.(q_2 \vee \langle 1 \rangle Z)$ 和 $\mu Z.(\neg q_2 \vee [1]Z)$ 且模型不满足 $\nu Z.(\neg q_2 \wedge [1]Z)$ 。

例 6.4 (1) CTL公式 $AGEFp$ 不能用LTL公式表示。

证: 定义 $M = \langle S, R, I, L \rangle$ 其中

$$\begin{aligned}
S &= \{s_0, s_1\}, \\
R &= \{(s_0, s_0), (s_0, s_1), (s_1, s_1)\}, \\
I &= \{s_0\}, \\
L(s_0) &= \{\}, L(s_1) = \{p\}.
\end{aligned}$$

定义 $M' = \langle S', R', I', L' \rangle$ 其中 $S' = \{s_0\}$, $R' = \{(s_0, s_0)\}$, $I' = \{s_0\}$, $L'(s_0) = \{\}$ 。

我们有 $M \models AGEFp$ 且 $M' \not\models AGEFp$ 。

设 $AGEFp$ 能用LTL公式 φ 表示。则 $M \models \varphi$ 且 $M' \not\models \varphi$ 。

根据语义, 由于 M' 的计算是 M 的计算的一个子集, 我们有 $M \models \varphi$ 蕴含 $M' \models \varphi$, 与 $M' \not\models \varphi$ 矛盾。

因而 $AGEFp$ 不能用LTL公式表示。

(2) LTL公式 $F(p \wedge Xp)$ 不能用CTL公式表示。

证: 定义 $M_1, N_1, M_{i+}, N_{i+1}$ 如下。

$$\begin{aligned}
M_1 &= \langle S_1, R_1, I_1, L_1 \rangle \text{ 其中} \\
S_1 &= \{s_2, s_1, s_0\}, \\
R_1 &= \{(s_2, s_1), (s_1, s_0), (s_0, s_0)\}, \\
I_1 &= \{s_2\}, \\
L_1(s_2) &= L_1(s_1) = \{p\} \text{ 且 } L_1(s_0) = \{\}. \\
N_1 &= \langle S'_1, R'_1, I'_1, L'_1 \rangle \text{ 其中} \\
S'_1 &= \{s'_2, s'_1\}, \\
R'_1 &= \{(s'_2, s'_1), (s'_1, s'_1)\}, \\
I'_1 &= \{s'_2\}, \\
L'_1(s'_2) &= \{p\} \text{ 且 } L'_1(s'_1) = \{\}. \\
M_{i+1} &= \langle S_{i+1}, R_{i+1}, I_{i+1}, L_{i+1} \rangle \text{ 其中} \\
S_{i+1} &= S_i \cup S'_i \cup \{s_{2i+2}, s_{2i+1}\}, \\
R_{i+1} &= R_i \cup R'_i \cup \{(s_{2i+2}, s_{i+1}), (s_{2i+1}, s_{2i}), (s_{2i+2}, s_{2i}), (s_{2i+2}, s'_{2i})\}, \\
I_{i+1} &= \{s_{2i+2}\}, \\
L_{i+1} &= L_i \cup L'_i \cup \{(s_{2i+2}, \{p\}), (s_{2i+1}, \{\})\}. \\
N_{i+1} &= \langle S'_{i+1}, R'_{i+1}, I'_{i+1}, L'_{i+1} \rangle \text{ 其中} \\
S'_{i+1} &= S_i \cup S'_i \cup \{s'_{2i+2}, s'_{2i+1}\}, \\
R'_{i+1} &= R_i \cup R'_i \cup \{(s'_{2i+2}, s'_{i+1}), (s'_{2i+1}, s'_{2i}), (s'_{2i+2}, s'_{2i}), (s'_{2i+2}, s'_{2i})\}, \\
I'_{i+1} &= \{s'_{2i+2}\}, \\
L'_{i+1} &= L_i \cup L'_i \cup \{(s'_{2i+2}, \{p\}), (s'_{2i+1}, \{\})\}.
\end{aligned}$$

我们有 $M_{i+1} \models F(p \wedge Xp)$ 且 $N_{i+1} \not\models F(p \wedge Xp)$ 。

考虑 CTL 算子 EX, EG, EU 。设 φ 为仅用 $\{EX, EG, EU\}$ 算子和命题逻辑连接符构造的 CTL 公式。

若 φ 的长度为 i ，则对所有 $k \geq i$ 有 $M_k \models \varphi$ 当且仅当 $N_k \models \varphi$ 。

因而 $F(p \wedge Xp)$ 不能用 CTL 公式表示。

§6.5 练习

1. 分别用 CTL 语义和推理系统证明 $AG(p \rightarrow AXp) \rightarrow (p \rightarrow AGp)$ 。
2. 设 K 如例 6.1 中所定义。用限界语义验证该模型是否满足 $A(q_0 U q_1)$ 和 $EG(q_0 \vee q_1)$ 并讨论最小的可以确定模型是否满足以上公式的界。
3. 设 K 如例 6.1 中所定义。用不动点算法计算 $[[A(q_0 U q_1)]]$ 和 $[[EG(q_0 \vee q_1)]]$ 并讨论模型是否满足 $A(q_0 U q_1)$ 和 $EG(q_0 \vee q_1)$ 。
4. 设 K' 如例 6.3 中所定义。用不动点算法计算 $\mu Z.(q_1 \vee \langle 1 \rangle Z)$ 和 $\mu Z.(q_1 \vee \langle 2 \rangle Z)$ 并讨论模型是否满足 $\mu Z.(q_1 \vee \langle 1 \rangle Z)$ 和 $\mu Z.(q_1 \vee \langle 2 \rangle Z)$ 。
5. 讨论满足公式 $A((p \vee q)Ur)$, $A((pUr) \vee (qUr))$ 与 $A(pUr) \vee A(qUr)$ 的模型的特点、指出这些公式之间的强弱关系。

§7 基于演绎推理的验证方法

本章介绍验证变量迁移模型是否具备给定性质的演绎推理方法(Deductive Reasoning)。本章的模型以一阶逻辑为基础。以下假定 $B = (F, P)$ 以及 B 的解释 $I = (D, I_0)$ 为给定。

§7.1 卫式迁移模型的推理

本节介绍卫式迁移模型的一阶线性时序逻辑性质的基于演绎推理的验证方法。给定变量集合 V 。设 $M = (T, \Theta)$ 为给定的 (B, V) 上的卫式迁移模型。设 B 的解释 I 为给定。本节用 φ, ψ 表示一阶逻辑公式。

定义 7.1 设 θ 为一阶线性时序逻辑公式。模型 M 满足 θ ，记作 $M \models_I \theta$ ，当且仅当 M 的行为满足 θ ，即 $[[M]] \models \theta$ 。

为方便起见，我们用 $\varphi \Rightarrow \psi$ 表示 $M \models_I \Box(\varphi \rightarrow \psi)$ 。

以下考虑形如 $\varphi_0 \Rightarrow (\varphi_1 U \varphi_2)$ 和 $\varphi_0 \Rightarrow (\varphi_1 R \varphi_2)$ 的一阶时序逻辑性质的验证。

命题 7.1 φ 是 M 的安全性质，当且仅当 $M \models_I \Box\varphi$ 。

命题 7.2 φ 是 M 的必达性质，当且仅当 $M \models_I \Diamond\varphi$ 。

断言与最弱宽松前断言：

定义 7.2 设 S 为迁移集合。定义 $\sigma \xrightarrow{S} \sigma'$ 当且仅当存在 $t \in S$ 使得 $\sigma \xrightarrow{t} \sigma'$ 。

定义 7.3 设 S 为迁移集合。定义 $\models_I \{\varphi\}S\{\psi\}$ 当且仅当 $I(\varphi)(\sigma) \rightarrow \forall \sigma'. ((\sigma \xrightarrow{S} \sigma') \rightarrow I(\psi)(\sigma'))$ 。

命题 7.3 设 $Z = X \cup Y$ 为迁移集合。则 $\models_I \{\varphi\}Z\{\psi\}$ 当且仅当 $\models_I \{\varphi\}X\{\psi\}$ 且 $\models_I \{\varphi\}Y\{\psi\}$ 。

定义 7.4 设 S 为迁移集合。公式 φ 称为 (S, ψ) 的最弱宽松前断言，记作 $\varphi = wlp(S, \psi)$ ，当且仅当

$$I(\varphi)(\sigma) \leftrightarrow \forall \sigma'. ((\sigma \xrightarrow{S} \sigma') \rightarrow I(\psi)(\sigma'))。$$

命题 7.4 设 S 为迁移集合。 $\varphi = wlp(S, \psi)$ 当且仅当 $\models_I \{\varphi\}S\{\psi\}$ 且若 $\models_I \{\varphi'\}S\{\psi\}$ 则 $\varphi' \rightarrow \varphi$ 。

命题 7.5 设 S 为迁移集合。 $\models_I \{\varphi\}S\{\psi\}$ 当且仅当 $\varphi \rightarrow wlp(S, \psi)$ 。

命题 7.6 设 $Z = X \cup Y$ 为迁移集合。则 $wlp(Z, \psi) \leftrightarrow wlp(X, \psi) \wedge wlp(Y, \psi)$

最弱宽松前断言的计算： 设 S 为迁移集合。定义 $[S]\psi$ 如下：

$$\begin{aligned} [\{\}] \psi &= \top \\ [p \rightarrow (x_1, \dots, x_n) := (e_1, \dots, e_n)] \psi &= (p \rightarrow \psi[x_1/e_1] \cdots [x_n/e_n]) \\ [S] \psi &= \bigwedge_{t \in S} [t] \psi \end{aligned}$$

命题 7.7 设 S 为迁移集合。 $wlp(S, \psi) \equiv [S]\psi$ 。

推论 7.1 设 S 为迁移集合。 $\models_I \{\varphi\}S\{\psi\}$ 当且仅当 $\varphi \rightarrow [S]\psi$ 。

时序算子 O :

命题 7.8 $\varphi \Rightarrow O\psi$ 当且仅当对所有可达状态 σ , $I(\varphi)(\sigma) \rightarrow \forall \sigma'. ((\sigma \xrightarrow{M} \sigma') \rightarrow I(\psi)(\sigma'))$ 。

定义 7.5 设 S 为迁移集合, $t \in S$ 为 $p \rightarrow (x_1, \dots, x_n) := (e_1, \dots, e_n)$ 。定义 $E(t) = p \wedge E(S) = \bigvee_{t \in S} E(t)$ 。

定义 $T^+ = T \cup \{\neg E(T) \rightarrow (x_1) := (x_1)\}$ 。

命题 7.9 $\varphi \Rightarrow O\psi$ 当且仅当对所有可达状态 σ , $I(\varphi)(\sigma) \rightarrow \forall \sigma'. ((\sigma \xrightarrow{T^+} \sigma') \rightarrow I(\psi)(\sigma'))$ 。

命题 7.10 $\varphi \Rightarrow O\psi$ 当且仅当 $\varphi \Rightarrow wlp(T^+, \psi)$

推论 7.2 $\varphi \Rightarrow O\psi$ 当且仅当 $\varphi \Rightarrow [T^+]\psi$ 。

我们有 $[T^+]\psi = ([T]\psi) \wedge (E(T) \vee \psi)$ 。

推论 7.3 $\varphi \Rightarrow O\psi$ 当且仅当 $\varphi \Rightarrow [T]\psi$ 且 $\varphi \Rightarrow (E(T) \vee \psi)$ 。

初始条件 Θ :

命题 7.11 设 θ 为一阶线性时序逻辑公式。 $M \models_I \theta$ 当且仅当 $M \models_I \Theta \rightarrow \theta$ 。

推论 7.4 若 $\Theta \rightarrow \varphi$ 且 $\varphi \Rightarrow \Box\psi$, 则 ψ 是 M 的安全性质。

推论 7.5 若 $\Theta \rightarrow \varphi$ 且 $\varphi \Rightarrow \Diamond\psi$, 则 ψ 是 M 的必达性质。

推理规则: 综合以上内容, 我们有以下推理规则, 包括迁移规则, 迁移集合规则, 下一时刻规则, 推论规则, 初始状态规则。前两个规则针对迁移和迁移集合, 后三个规则针对给定模型 $M = (T, \Theta)$ 。

$\frac{\varphi \wedge p \rightarrow \psi[x_1/e_1] \cdots [x_n/e_n]}{\varphi \rightarrow [\{p \rightarrow (x_1, \dots, x_n) := (e_1, \dots, e_n)\}]\psi}$	$\frac{\varphi \rightarrow [S_0]\psi \quad \varphi \rightarrow [S_1]\psi}{\varphi \rightarrow [S_0 \cup S_1]\psi}$
$\frac{\varphi \rightarrow (E(T) \vee \psi) \quad \varphi \rightarrow [T]\psi}{\varphi \Rightarrow O\psi}$	$\frac{\varphi' \Rightarrow \varphi \quad \varphi \Rightarrow O\psi \quad \psi \Rightarrow \psi'}{\varphi' \Rightarrow O\psi'}$
$\frac{M \models_I \Theta \rightarrow \theta}{M \models_I \theta}$	

导出规则: 设 $e \in T_B$ 为项, \sqsubseteq 为 P 中二元谓词符号, $w \in QFF_B$ 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, v 为没有在公式中出现过的变量。结合一阶线性时序逻辑的推理规则和以上与模型相关的推理规则, 我们有以下规则。

$\varphi \Rightarrow \phi$	$\varphi \Rightarrow \phi$
$(\phi \wedge \neg\varphi_0) \rightarrow [T]\phi$	$\phi \rightarrow [T]\phi$
$\phi \rightarrow \psi$	$\phi \rightarrow \psi$
<hr/>	<hr/>
$\varphi \Rightarrow \varphi_0 R\psi$	$\varphi \Rightarrow \Box\psi$
$\varphi \Rightarrow (\psi \vee \phi)$	$\varphi \Rightarrow (\psi \vee \phi)$
$\phi \rightarrow (\varphi_0 \wedge w_x^e \wedge (E(T) \vee \psi))$	$\phi \rightarrow (w_x^e \wedge (E(T) \vee \psi))$
$(\phi \wedge e = v) \rightarrow [T](\psi \vee (\phi \wedge e \sqsubset v))$	$(\phi \wedge e = v) \rightarrow [T](\psi \vee (\phi \wedge e \sqsubset v))$
<hr/>	<hr/>
$\varphi \Rightarrow \varphi_0 U\psi$	$\varphi \Rightarrow \Diamond\psi$

对于推理规则的使用，我们需要构造推理规则前提中使用的 e 和 w 等元素。若 $B = (F, P)$ 的表达能力不够，从实际应用考虑，可对 B 进行适当扩展和增加相应解释，以便于定义 e 和 w 等元素。

§7.2 流程图模型的推理

对于流程图模型，传统的性质包括部分正确和完全正确。这些性质约束的是模型运行终止时程序的变量状态，以及模型运行是否能够终止。本节介绍流程图模型的这些正确性问题的基于演绎推理的验证方法。

设 T 为给定的 (B, V) 上的流程图模型。

定义模型的语义函数为变量状态到变量状态的偏函数 $\mathcal{M}_T(T) : \Sigma \hookrightarrow \Sigma$ 如下。

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

$\mathcal{M}_T(T)(\sigma)$ 有定义或者说模型 T 在初始变量状态为 σ 时的运行终止记作 $\mathcal{M}_T(T)(\sigma) \downarrow$ 。设 p 和 q 是两个公式。

- 部分正确性: $\models_I \{p\}T\{q\}$ 当且仅当 $\forall \sigma. (I(p)(\sigma) \rightarrow (\mathcal{M}_T(T)(\sigma) \downarrow \rightarrow I(q)(\mathcal{M}_T(T)(\sigma))))$;
- 终止性: $\models_I [p]T[true]$ 当且仅当 $\forall \sigma. (I(p)(\sigma) \rightarrow (\mathcal{M}_T(T)(\sigma) \downarrow))$ 。
- 完全正确性: $\models_I [p]T[q]$ 当且仅当 $\forall \sigma. (I(p)(\sigma) \rightarrow (\mathcal{M}_T(T)(\sigma) \downarrow \wedge I(q)(\mathcal{M}_T(T)(\sigma))))$ 。

§7.2.1 直接证明

对于部分正确要求而言(设前置谓词和后置谓词分别为 φ 和 ψ)，证明模型满足部分正确的一种思路是假设模型的运行行为

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

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

对于终止而言(设前置谓词为 φ)，一种证明思路是假设模型的运行不终止

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

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。

如果直接证明完全正确性, 那么一种思路是首先假设模型的运行行为

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

然后证明若 $\varphi(\sigma_0)$ 成立则通过 n 次 (n 与初始变量状态 σ_0 相关) 状态变化后, 我们有 $l_n = end$ 且 $\psi(\sigma_n) = true$ 。

§7.2.2 基于路径的推理

以上的证明思路是基于对模型运行过程的分析。对于简单的程序, 该思路是可行的。对于复杂的模型, 一个重要的原则是将一个大的问题分解成小的问题。对流程图模型而言, 我们可以将模型的运行分成不同的片段, 由这些片段的正确性推导整个模型的正确性。模型片段用标号序列刻画。称标号序列为路径。

定义 7.6 给定模型 T 。标号序列 (l_0, l_1, \dots, l_n) 是 T 的路径当且仅当 $n > 0$ 且对所有 $0 \leq i < n$, l_{i+1} 在 l_i 的定义中出现。

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

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

$\mathcal{M}_I(\alpha)(\sigma)$ 有定义或者说路径 α 在初始变量状态为 σ 时的运行完成记作 $\mathcal{M}_I(\alpha)(\sigma) \downarrow$ 。

命题 7.12 若 $l_0 = beg$ 且 $l_n = end$ 则 $\alpha = (l_0, l_1, \dots, l_n)$ 称为完整路径。 $\mathcal{M}_I(T)(\sigma) = \sigma'$ 当且仅当存在完整路径 α 使得 $\mathcal{M}_I(\alpha)(\sigma) = \sigma'$ 。

用 $\sigma \xrightarrow{\alpha} \sigma'$ 表示 $\mathcal{M}_I(\alpha)(\sigma) \downarrow \wedge \sigma' = \mathcal{M}_I(\alpha)(\sigma)$ 。设 α 为路径, p 和 q 为公式。类似于模型正确性, 我们定义路径的部分正确性。

定义 7.7 α 在 I 解释下对于 p 和 q 部分正确, 记作 $\models_I \{p\}\alpha\{q\}$, 若 $\forall \sigma. (I(p)(\sigma) \rightarrow ((\sigma \xrightarrow{\alpha} \sigma') \rightarrow I(q)(\sigma')))$ 。

推论 7.6 $\models_I \{p\}T\{q\}$ 当且仅当对所有完整路径 α 有 $\models_I \{p\}\alpha\{q\}$ 。

定义 7.8 设 α 为路径, p 和 q 是两个公式。 p 称为 α 和 q 的最弱宽松前断言, 记作 $p = wlp(\alpha, q)$, 当且仅当 $I(p)(\sigma) \leftrightarrow ((\sigma \xrightarrow{\alpha} \sigma') \rightarrow I(q)(\sigma'))$ 。

命题 7.13 $p \equiv wlp(\alpha, q)$ 当且仅当 $\models_I \{p\}\alpha\{q\}$ 且若 $\models_I \{p'\}\alpha\{q\}$ 则 $p' \rightarrow p$ 。

命题 7.14 $\models_I \{p\}\alpha\{q\}$ 当且仅当 $p \rightarrow wlp(\alpha, q)$ 。

设 $\alpha = (l_1, \dots, l_k), \beta = (l'_1, \dots, l'_m)$ 为路径。若 $l_k = l'_1$, 则称 α 与 β 为可连接路径, 记 $\alpha \cdot \beta$ 为由 α 与 β 连接生成的路径 $(l_1, \dots, l_{k-1}, l'_1, \dots, l'_m)$ 。

命题 7.15 设 $\gamma = \alpha \cdot \beta$ 为路径。若 $\models_I \{p\}\alpha\{q\}$ 且 $\models_I \{q\}\beta\{r\}$, 则 $\models_I \{p\}\gamma\{r\}$ 。

命题 7.16 设 $\gamma = \alpha \cdot \beta$ 为路径。 $wlp(\gamma, q) \equiv wlp(\alpha, wlp(\beta, q))$ 。

最弱宽松前断言的计算： 设 $\alpha = (l_0, \dots, l_k)$ 为路径。定义 $[\alpha]q$ 如下。

$k = 1$	T 中存在指令 $l_0: (v_1, \dots, v_n) := (e_1, \dots, e_n) \text{ goto } l_1$ $[\alpha]q = q[v_1/e_1] \cdots [v_n/e_n]$.
	T 中存在指令 $l_0: \text{if } (b) \text{ goto } l \text{ else goto } l' \text{ fi}$ $[\alpha]q = \begin{cases} b \rightarrow q & \text{若 } l = l_1; \\ \neg b \rightarrow q & \text{若 } l' = l_1. \end{cases}$
$k > 1$	$[\alpha]q = [(l_0, \dots, l_{k-1})][(l_{k-1}, l_k)]q$

命题 7.17 $wlp(\alpha, q) \equiv [\alpha]q$ 。

推论 7.7 $\models_I \{p\}\alpha\{q\}$ 当且仅当 $p \rightarrow [\alpha]q$ 。

$p \rightarrow [\alpha]q$ 称为 $\{p\}\alpha\{q\}$ 的验证条件，记为 $vc(p, \alpha, q)$ 。

基于归纳断言的部分正确性证明：

定义 7.9 设 T 为模型， C 为标号集合。定义 $\gamma_T(C)$ 如下：

$(l_0, \dots, l_k) \in \gamma_T(C)$ 当且仅当 (l_0, \dots, l_k) 是 T 的路径， $l_0, l_k \in C$ 且 $l_1, \dots, l_{k-1} \notin C$ 。

命题 7.18 设 C 是标号集合， $beg, end \in C$ ， T 的每个循环至少有一个标号包含于 C 且 C 中的每个标号 l 有一个对应的公式 q_l 。若 $\forall \alpha = (l_0, \dots, l_k) \in \gamma_T(C), \models_I q_{l_0} \rightarrow [\alpha]q_{l_k}$ ，则 $\models \{q_{beg}\}T\{q_{end}\}$ 。

终止性的特点： 终止性不是一阶性质。以下模型在一阶算术 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

```

基于谓词的终止性证明：

命题 7.19 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 。 $(W, \sqsubseteq, I_0(\sqsubseteq))$ 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) $\forall \alpha = (l_0, \dots, l_k) \in \gamma_T(C), \models_I q_{l_0} \rightarrow [\alpha]q_{l_k}$ |
| (b) $\forall \alpha = (l_0, \dots, l_k) \in \gamma_T(C), I(q_{l_0})(\sigma) \wedge (\sigma \xrightarrow{\alpha} \sigma') \rightarrow g_{l_k}(\sigma') \sqsubseteq q_{l_0}(\sigma)$ |

基于谓词公式的终止性证明：

命题 7.20 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 。 $(W \subseteq D, I_0(\sqsubseteq))$ 是一良基集合且 $\sqsubseteq \in P$ 。 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) $\forall \alpha = (l_0, \dots, l_k) \in \gamma_T(C), \models_I q_{l_0} \rightarrow [\alpha]q_{l_k}$ |
| (b) $\forall \alpha = (l_0, \dots, l_k) \in \gamma_T(C), \models_I q_{l_0} \wedge t_{l_0} = v \rightarrow [\alpha](t_{l_k} \sqsubseteq v)$ |

这个方法与上述方法相比有一定的局限性，即良基集合的选择受限于 $W \subseteq D$ 且谓词符号必须在 P 中。这个方法的优点是推理过程简单，不用直接考虑状态的内容。

§7.3 结构化程序模型的推理

流程图模型的一个缺点是循环的入口和出口不规范，从证明的角度讲，推理比较复杂。从模型结构来讲，可组合性较差。结构化程序模型可以克服这些缺点。为方便叙述，亦称结构化程序模型为结构化程序。本节介绍结构化程序模型的部分正确性与终止性质的基于演绎推理的验证方法。

设 T 为给定的 (B, V) 上的结构化程序。

定义结构化程序的语义函数为变量状态到变量状态的偏函数 $\mathcal{M}_T(T) : \Sigma \hookrightarrow \Sigma$ 如下。

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

$\mathcal{M}_T(T)(\sigma)$ 有定义或者说结构化程序 T 在初始变量状态为 σ 时的运行终止记作 $\mathcal{M}_T(T)(\sigma) \downarrow$ 。设 p 和 q 是两个公式。

- 部分正确性: $\models_I \{p\}T\{q\}$ 当且仅当 $\forall \sigma. (I(p)(\sigma) \rightarrow (\mathcal{M}_T(T)(\sigma) \downarrow \rightarrow I(q)(\mathcal{M}_T(T)(\sigma))))$;
- 终止性: $\models_I [p]T[true]$ 当且仅当 $\forall \sigma. (I(p)(\sigma) \rightarrow (\mathcal{M}_T(T)(\sigma) \downarrow))$ 。
- 完全正确性: $\models_I [p]T[q]$ 当且仅当 $\forall \sigma. (I(p)(\sigma) \rightarrow (\mathcal{M}_T(T)(\sigma) \downarrow \wedge I(q)(\mathcal{M}_T(T)(\sigma))))$ 。

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

§7.3.1 指称语义

结构化程序具有组合性。比如给定结构化程序 T_1 和 T_2 ，我们可将其组合成 $T_1; T_2$ ，再给一个公式 e ，我们可将其组合成 $\text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}$ 等。这样， $T_1; T_2$ 的语义就可以建立在 T_1 和 T_2 的语义上。比如 $\mathcal{M}_T(T_1; T_2)(\sigma) = \mathcal{M}_T(T_2)\mathcal{M}_T(T_1)(\sigma)$ 。由于 $\mathcal{M}_T(T_1)(\sigma)$ 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^ω 。给定结构化程序 T ， $\mathcal{M}_I^\omega(T)$ 是 Σ_ω 上的函数。

定义 $\text{ite} : \text{Bool} \times \Sigma_\omega \times \Sigma_\omega \rightarrow \Sigma_\omega$ 如下

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

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

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

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

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

对于任意结构化程序 T , $\mathcal{M}_I^\omega(T)$ 是良定义且连续的。 $\mathcal{M}_I^\omega(T)$ 的定义和之前 $\mathcal{M}_I(T)$ 的定义是一致的, 即我们有如下结论。

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

反过来, 我们有

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

部分正确性: 部分正确性的要求证明 $\varphi(\sigma) \rightarrow (\mathcal{M}_I(T)(\sigma) \downarrow \rightarrow \psi(\mathcal{M}_I(T)(\sigma)))$ 。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运行一定终止, 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 $\varphi(\sigma) \rightarrow \psi(h(\sigma))$ 。

循环语句的特殊处理: 应用指称语义进行证明, 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 $\varphi: \Sigma^2 \rightarrow \text{Bool}$ 为谓词。设 $T = \text{while } e \text{ do } T_1 \text{ od}$ 。若以下命题成立, 则 $\forall \sigma \in \Sigma, \mathcal{M}_I(T)(\sigma) \downarrow \rightarrow \varphi(\sigma, \mathcal{M}_I(T)(\sigma))$ 。

$$\begin{array}{l} \forall \sigma \in \Sigma, I(\neg e)(\sigma) \rightarrow \varphi(\sigma, \sigma) \\ \forall \sigma, \sigma' \in \Sigma, I(e)(\sigma) \wedge \mathcal{M}_I(T_1)(\sigma) \downarrow \wedge \varphi(\mathcal{M}_I(T_1)(\sigma), \sigma') \rightarrow \varphi(\sigma, \sigma') \end{array}$$

这里我们只计算 $\mathcal{M}_I(T_1)(\sigma)$ 而避开了不动点, 因此应用该方法可简化一些循环语句正确性的证明。

终止性: 结构化程序 T 的终止性可以根据已知结构化程序 T' 的终止性来证明。若 $\mathcal{M}_I(T')(\sigma)$ 为处处有定义的函数且 $\mathcal{M}_I(T)(\sigma) \sqsubseteq \mathcal{M}_I(T')(\sigma)$ 则 $\mathcal{M}_I(T)(\sigma)$ 处处有定义。

§7.3.2 Hoare逻辑

除了基于指称语义的证明, 我们还可以用逻辑的方法来证明结构化程序的性质。Hoare逻辑就是这样的一个可以用来证明结构化程序性质的系统。Hoare逻辑语言建立在谓词逻辑和结构化程序的基础上。

定义 7.10 Hoare公式的定义如下。

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

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

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

$I(\{p\}T\{q\})(\sigma) = true$ 当且仅当 $I(p)(\sigma) = true \wedge M_I(T)(\sigma) \downarrow \rightarrow I(q)(M_I(T)(\sigma)) = true.$

一个公式 $\{p\}T\{q\}$ 在 I 的解释下成立, 即对任意 σ , $I(\{p\}T\{q\})(\sigma) = true$, 记为 $\models_I \{p\}T\{q\}$ 。若其在任意解释下成立, 记为 $\models \{p\}T\{q\}$ 。若其在满足 W 的解释下成立, 记为 $W \models \{p\}T\{q\}$ 。

最弱宽松前断言:

定义 7.11 设 T 为程序片段, p 和 q 是两个公式。 p 称为 T 和 q 的最弱宽松前断言, 记作 $p = wlp(T, q)$, 当且仅当 $I(p)(\sigma) \leftrightarrow (M_I(T)(\sigma) \downarrow \rightarrow I(q)(M_I(T)(\sigma)))$ 。

命题 7.21 $p \equiv wlp(T, q)$ 当且仅当 $\models_I \{p\}T\{q\}$ 且若 $\models_I \{p'\}T\{q\}$ 则 $p' \rightarrow p$ 。

命题 7.22 $\models_I \{p\}T\{q\}$ 当且仅当 $p \rightarrow wlp(T, q)$ 。

假定我们使用的逻辑具有足够强的表达能力且以下定义中涉及的 r 可以用逻辑公式表达。

定义 7.12 设 T 为程序片段, q 是公式。定义 $[T]q$ 如下。

T	$[T]q$
$v := e$	$q(v/e)$
$T_1; T_2$	$[T_1][T_2]q$
$if(b) then T_1 else T_2 fi$	$(b \rightarrow [T_1]q) \wedge (\neg b \rightarrow [T_2]q)$
$while(b) do T_1 od$	r

其中 r 满足 $r \rightarrow (b \rightarrow [T_1]r) \wedge (\neg b \rightarrow q)$ 且若 $q' \rightarrow (b \rightarrow [T_1]q') \wedge (\neg b \rightarrow q)$ 则 $q' \rightarrow r$ 。

命题 7.23 $wlp(T, q) \equiv [T]q$ 。

推论 7.8 $\models_I \{p\}T\{q\}$ 当且仅当 $p \rightarrow [T]q$ 。

推理规则： 设 $x \in V$, $t \in T_B$, 设 $e \in QFF_B$, $p, q, r, s \in WFF_B$ 且 $T, T_1, T_2 \in \mathcal{L}_O^{(B,V)}$ 。我们有以下公理和推理规则。

(1) 赋值公理：

$$\{p[t/x]\}x := t\{p\}$$

(2) 组合规则：

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

(3) 条件规则：

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

(4) 循环规则：

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

(5) 推论规则：

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

一个Hoare公式 $\{p\}T\{q\}$ 可由以上规则和公理得到，记作 $\vdash \{p\}T\{q\}$ 。

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

为了专注于与结构化程序直接相关的性质的推理，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： 结构化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

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

我们有以下性质。

$$\begin{aligned} & \models_I \{q'_x\}x := t\{q\} \\ & \models_I \{p\}T_1\{r\} \text{ 且 } \models_I \{r\}T_2\{q\}, \text{ 则 } \models_I \{p\}T_1; T_2\{q\} \\ & \models_I \{p \wedge e\}T_1\{q\} \text{ 且 } \models_I \{p \wedge \neg e\}T_2\{q\}, \text{ 则 } \models_I \{p\}\text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}\{q\} \\ & \models_I \{p \wedge e\}T_1\{p\}, \text{ 则 } \models_I \{p\} \text{ while } e \text{ do } T_1 \text{ od}\{p \wedge \neg e\} \\ & \models_I \{q\}T\{r\}, \models_I p \rightarrow q \text{ 且 } \models_I r \rightarrow s, \text{ 则 } \models_I \{p\}T\{q\} \end{aligned}$$

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：

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

我们有以下性质。

$$\begin{array}{l} \hline \vdash_I \{p\}x := t\{q\}, \text{ 则 } \vdash_I p \rightarrow q'_x \\ \vdash_I \{p\}T_1; T_2\{q\} \text{ 且 } I(r) \text{ 为 } T_2 \text{ 和 } I(q) \text{ 的最弱宽松前断言, 则 } \vdash_I \{p\}T_1\{r\} \text{ 且 } \vdash_I \{r\}T_2\{q\} \\ \vdash_I \{p\} \text{if } e \text{ then } T_1 \text{ else } T_2 \text{ fi}\{q\}, \text{ 则 } \vdash_I \{p \wedge e\}T_1\{q\} \text{ 且 } \vdash_I \{p \wedge \neg e\}T_2\{q\} \\ \vdash_I \{p\} \text{while } e \text{ do } T_1 \text{ od}\{q\}, \text{ 则 } \vdash_I \{p\} \text{if } (e) T_1; \text{while } e \text{ do } T_1 \text{ od else } x := x \text{ fi}\{q\} \\ \hline \end{array}$$

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

给定 I 且假定 I 的表达能力强 enough，对 T 做结构归纳，我们有

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

即Hoare逻辑推理系统具有相对完备性。

导出规则： 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。若用已有的公理和规则可以从公式 s_1, s_2, \dots, s_n 推导出 s ，则可以产生以下导出规则。

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

设 $x \in V$, $t \in T_B$, $e \in QFF_B$, $p, q, r, p_0, \dots, p_n \in WFF_B$ 且 $T_1, \dots, T_n \in \mathcal{L}_{\circ}^{(B,V)}$ 。以下是一些常用的导出规则。这些规则可用以替代原有的赋值公理，组合规则，条件规则和循环规则。

(1')

$$\frac{p \rightarrow q'_x}{\{p\}x := t\{q\}}$$

(2')

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

(3')

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

(4')

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

循环规则的替代规则： 循环规则中的 r 称为循环不变式。应用Hoare逻辑进行证明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 $T = \text{while } e \text{ do } T_1 \text{ od}$ 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 $p, q, r \in WFF_B, e \in QFF, T_1 \in \mathcal{L}_{\circ}^{(B,V)}$ 。 则

$$\frac{\neg e \rightarrow r_{x'_1, \dots, x'_n}^{x_1, \dots, x_n}, \{\neg r \wedge e\} T_1 \{\neg r\}, (p \wedge r) \rightarrow q_{x'_1, \dots, x'_n}^{x_1, \dots, x_n}}{\{p\} \text{while } e \text{ do } T_1 \text{ od} \{q\}}$$

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。

§7.3.3 Hoare逻辑的扩展

Hoare逻辑只证明结构化程序的部分正确，不证明程序的终止性和完全正确。为了证明终止性和完全正确，需要对Hoare逻辑进行扩展。给定 (B, V) 。

定义 7.13 扩展Hoare公式的定义如下。

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

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

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

$I([p]T[q])(\sigma) = true$ <p>当且仅当</p> $I(p)(\sigma) = true \rightarrow M_I(T)(\sigma) \downarrow \wedge I(q)(M_I(T)(\sigma)) = true.$

最弱前断言：

定义 7.14 设 T 为程序片段， p 和 q 是两个公式。 p 称为 T 和 q 的最弱前断言，记作 $p = wp(T, q)$ ，当且仅当 $I(p)(\sigma) \leftrightarrow (M_I(T)(\sigma) \downarrow \wedge I(q)(M_I(T)(\sigma)))$ 。

命题 7.24 $p \equiv wp(T, q)$ 当且仅当 $\models_I [p]T[q]$ 且若 $\models_I [p']T[q]$ 则 $p' \rightarrow p$ 。

命题 7.25 $\models_I [p]T[q]$ 当且仅当 $p \rightarrow wp(T, q)$ 。

结构化程序的运行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良基集合。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 $(W, I_0(\leq))$ 为一良基集合。
 - (3) 存在 $w \in WFF_B$ 且 w 包含不多于一个变量，记为 x ，使得 $W = \{\sigma(x) \mid I(w)(\sigma) = true\}$ 。
- 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，条件语句和推论规则，依照Hoare逻辑推理规则的形式把 $\{p\}T\{q\}$ 替换成 $[p]T[q]$ 即可。

§7.4 例子

例 7.1 设 t_1, t_2 分别为以下迁移。

$$\begin{aligned} t_1 &: \top \longrightarrow (x) := (x + 1) \\ t_2 &: y = 1 \longrightarrow (x) := (x + 2) \end{aligned}$$

(1) 以下成立。

$$\begin{aligned} &\models \{x > 1\}\{t_1\}\{x > 2\} \\ &\models \{y = 1 \rightarrow x > 1\}\{t_2\}\{x > 2\} \\ &\models \{x > 1\}\{t_1, t_2\}\{x > 2\} \end{aligned}$$

(2) 以下成立。

$$\begin{aligned} wlp(\{t_1\}, x > 2) &= x > 1 \\ wlp(\{t_2\}, x > 2) &= y = 1 \rightarrow x > 0 \\ wlp(\{t_1, t_2\}, x > 2) &= x > 1 \end{aligned}$$

(3) 以下成立。

$$\begin{aligned} [\{t_1\}](x > 2) &= x > 1 \\ [\{t_2\}](x > 2) &= y = 1 \rightarrow x > 0 \\ [\{t_1, t_2\}](x > 2) &= x > 1 \end{aligned}$$

例 7.2 设 $B = (F, P)$ 其中 $F = \{s_0, s_1, s_2, t_0, t_1, t_2, 0, 1\}$ 且 $P = \{=\}$ 。设 $V = \{a, b, x, y, t\}$ 。定义 (T, Θ) 如下。

- T 为以下迁移的集合。

$$\begin{aligned} a = s_0 &\longrightarrow (a, y, t) := (s_1, 1, 1) \\ a = s_1 \wedge (x = 0 \vee t = 0) &\longrightarrow (a) := (s_2) \\ a = s_2 &\longrightarrow (a, y) := (s_0, 0) \\ b = t_0 &\longrightarrow (b, x, t) := (t_1, 1, 0) \\ b = t_1 \wedge (y = 0 \vee t = 1) &\longrightarrow (b) := (t_2) \\ b = t_2 &\longrightarrow (b, x) := (t_0, 0) \end{aligned}$$

- 设 Θ 为以下公式。

$$a = s_0 \wedge b = t_0 \wedge x = 0 \wedge y = 0$$

则 $M = (T, \Theta)$ 为 (B, V) 上的卫式迁移模型。

设 B 的解释 $I = (D, I_0)$ 定义如下。

- $D = \{0, 1, 2, 3\}$ 。
- I_0 定义如下。

$$\begin{aligned} I_0(0) &= 0 \\ I_0(1) &= 1 \\ I_0(s_i) = I_0(t_i) &= i \quad \text{for } i \in \{0, 1, 2\} \\ I_0(=) &= = \end{aligned}$$

用 u_1, \dots, u_6 分别代表 T 的6条迁移。则 $wlp(T, b \neq t_2)$ 的计算如下。

$$\begin{aligned} wlp(T, b \neq t_2) &= \\ [T](b \neq t_2) &= \\ [\{u_1\}](b \neq t_2) \wedge \dots \wedge [\{u_6\}](b \neq t_2) &= \\ (a = s_0 \rightarrow b \neq t_2) \wedge (a = s_1 \wedge (x = 0 \vee t = 0) \rightarrow b \neq t_2) \wedge (a = s_2 \rightarrow b \neq t_2) \wedge \\ (b = t_1 \rightarrow t_1 \neq t_2) \wedge (b = t_2 \wedge (y = 0 \vee t = 1) \rightarrow t_2 \neq t_2) \wedge (b = t_3 \rightarrow t_3 \neq t_2) &= \\ (a = s_0 \rightarrow b \neq t_2) \wedge (a = s_1 \wedge (x = 0 \vee t = 0) \rightarrow b \neq t_2) \wedge (a = s_2 \rightarrow b \neq t_2) \wedge \\ (\neg(b = t_2 \wedge (y = 0 \vee t = 1))) & \end{aligned}$$

模型满足 $(b \neq t_1 \wedge b \neq t_2) \Rightarrow O(b \neq t_2)$ 的证明如下。

根据推理规则

$$\frac{\begin{array}{l} \varphi \rightarrow (E(T) \vee \psi) \\ \varphi \rightarrow [T]\psi \end{array}}{\varphi \Rightarrow O\psi}$$

我们需证明 $(b \neq t_1 \wedge b \neq t_2) \Rightarrow (E(T) \vee b \neq t_2)$ 和 $(b \neq t_1 \wedge b \neq t_2) \Rightarrow [T](b \neq t_2)$ 。

第一个条件显然成立，由前面 $[T](b \neq t_2)$ 的计算结果易知第二个条件亦成立，因而 $(b \neq t_1 \wedge b \neq t_2) \Rightarrow O(b \neq t_2)$ 成立。

例 7.3 定义 (B, V) 上的卫式迁移模型 $M = (T, \Theta)$ 如下（类似于例3.2中 M'_3 ）。

$T :$	$pc = s_0$	\rightarrow	$(y_1, y_2, r, pc) := (1, 1, 1, s_1)$
	$pc = s_1 \wedge (y_2 \leq z)$	\rightarrow	$(pc) := (s_2)$
	$pc = s_1 \wedge \neg(y_2 \leq z)$	\rightarrow	$(pc) := (s_4)$
	$pc = s_2$	\rightarrow	$(y_1, y_2, pc) := (y_2, y_1 + y_2, s_3)$
	$pc = s_3$	\rightarrow	$(r, pc) := (r + 1, s_1)$
$\Theta :$	$(a = s_0)$		

设 $B = (F, P)$ 和 V 中的符号和变量足以表达以上出现的项和公式， I_0 为 B 中符号在通常自然数上的解释且 $I_0(fib) = fib$ 和 $I_0(bif) = bif$ 满足 $r = bif(z) \equiv fib(r) \leq z \wedge z < fib(r + 1)$ （见例3.2中的解释）。

性质 $M \models_I z > 0 \rightarrow \Box(a = s_4 \rightarrow r = bif(z))$ 的证明如下。

若有 $a = s_0 \wedge z > 0 \Rightarrow \Box(a = s_4 \rightarrow r = bif(z))$, 则有 $M \models_I a = s_0 \wedge z > 0 \rightarrow \Box(a = s_4 \rightarrow r = bif(z))$, 然后由 Θ 规则可得 $M \models_I z > 0 \rightarrow \Box(a = s_4 \rightarrow r = bif(z))$ 。

以下证明 $a = s_0 \wedge z > 0 \Rightarrow \Box(a = s_4 \rightarrow r = bif(z))$ 。

设 $\varphi = (a = s_0 \wedge z > 0)$, $\psi = (a = s_4 \rightarrow r = bif(z))$ 。

拟应用以下推理规则证明 $\varphi \Rightarrow \Box\psi$ 。

$$\begin{array}{l} \varphi \Rightarrow \phi \\ \phi \rightarrow [T]\phi \\ \phi \rightarrow \psi \\ \hline \varphi \Rightarrow \Box\psi \end{array}$$

我们需定义 ϕ 使得推理规则中的三个前提条件成立。

定义 ϕ 为 $(a = s_0 \wedge \phi_0) \vee (a = s_1 \wedge \phi_1) \vee (a = s_2 \wedge \phi_2) \vee (a = s_3 \wedge \phi_3) \vee (a = s_4 \wedge \phi_4)$ 其中

$$\begin{array}{l} \phi_0 \equiv (z > 0) \\ \phi_1 \equiv (r \geq 1 \wedge y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge y_1 \leq z) \\ \phi_2 \equiv (r \geq 1 \wedge y_2 = fib(r+1) \wedge y_1 + y_2 = fib(r+2) \wedge y_2 \leq z) \\ \phi_3 \equiv (r \geq 1 \wedge y_1 = fib(r+1) \wedge y_2 = fib(r+2) \wedge y_1 \leq z) \\ \phi_4 \equiv (r = bif(z)) \end{array}$$

推理规则中的第一个和第三个条件显然成立。通过计算 $[T]\phi$ 知第二个条件亦成立, 因而所需证明的结论成立。

(b) 性质 $M \models_I (z > 0 \rightarrow \Diamond(a = s_4))$ 的证明如下。

若有 $a = s_0 \wedge z > 0 \Rightarrow \Diamond(a = s_4)$, 则有 $M \models_I (a = s_0 \wedge z > 0 \rightarrow \Diamond(a = s_4))$, 然后由 Θ 规则可得 $M \models_I (z > 0 \rightarrow \Diamond(a = s_4))$ 。

以下证明 $a = s_0 \wedge z > 0 \Rightarrow \Diamond(a = s_4)$ 。

设 $\varphi = (a = s_0 \wedge z > 0)$, $\psi = (a = s_4)$ 。

拟应用以下推理规则证明 $\varphi \Rightarrow \Diamond\psi$ 。

$$\begin{array}{l} \varphi \Rightarrow (\psi \vee \phi) \\ \phi \rightarrow (w_x^e \wedge (E(T) \vee \psi)) \\ (\phi \wedge e = v) \rightarrow [T](\psi \vee (\phi \wedge e \sqsubset v)) \\ \hline \varphi \Rightarrow \Diamond\psi \end{array}$$

对 $B = (F, P)$ 进行扩充, 增加函数符号 h , 谓词符号 \sqsubset , 相应扩充 I_0 使得 \sqsubset 解释为通常自然数上的大小关系且 h 有如下解释。

$$\begin{array}{l} I(h(s_0, z, y_2)) = 3z + 1 \\ I(h(s_1, z, y_2)) = 3(z + 1 - y_2) \\ I(h(s_2, z, y_2)) = 3(z + 1 - y_2) - 1 \\ I(h(s_3, z, y_2)) = 3(z + 1 - y_2) + 1 \\ I(h(s_4, z, y_2)) = 0 \end{array}$$

我们需定义 e, w, ϕ 使得 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, I_0(\sqsubseteq))$ 为良基集合且推理规则中的三个前提条件成立。

设 ϕ_0, \dots, ϕ_4 如(a)中所定义。定义 w, e, ϕ 如下。

$$\begin{aligned} w &= true \\ e &= h(a, z, y_2) \\ \phi &= \bigvee_{i=0}^3 (pc = s_i \wedge \phi_i) \end{aligned}$$

由 w 的定义知 $W = \{0, 1, 2, \dots\}$ 为良基集合。

易知推理规则中的第一个条件和第二个条件成立。通过计算 $[T](\psi \vee (\phi \wedge e \sqsubseteq v))$ 知第三个条件亦成立，因而所需证明的结论成立。

证明中需要注意的是自然数中的减法与整数中的减法略有不同，即若 $a < b$ 则 $a - b = 0$ ，因而在论证 $3(z + 1 - (y_1 + y_2)) + 2 < 3(z + 1 - y_2)$ 时，除了说明 $y_1 \geq 1$ 外，还需要明确说明此时有 $y_2 \leq z$ 为前提条件。

(c) 设 I 为 B 中符号在通常整数上的解释。性质 $M \models_I (z > 0 \rightarrow \diamond(a = s_4))$ 的证明如下。

由以上(b)中的分析知证明 $a = s_0 \wedge z > 0 \Rightarrow \diamond(a = s_4)$ 即可。

设 $\varphi = (a = s_0 \wedge z > 0), \psi = (a = s_4)$ 。

应用的推理规则与(b)相同。证明的大部分构造与(b)中的构造类似。对 $B = (F, P)$ 进行扩充，增加函数符号 h' ，谓词符号 \sqsubseteq ，相应扩充 I_0 使得 \sqsubseteq 解释为通常整数上的大小关系且 h' 有如下解释。

$$\begin{aligned} I(h'(s_0, z, y_2)) &= 3z + 1 \\ I(h'(s_1, z, y_2)) &= \text{若 } y_2 \leq z \text{ 则 } 3(z + 1 - y_2) \text{ 否则 } 0 \\ I(h'(s_2, z, y_2)) &= \text{若 } y_2 \leq z \text{ 则 } 3(z + 1 - y_2) - 1 \text{ 否则 } 0 \\ I(h'(s_3, z, y_2)) &= \text{若 } y_2 \leq z \text{ 则 } 3(z + 1 - y_2) + 1 \text{ 否则 } 1 \\ I(h'(s_4, z, y_2)) &= 0 \end{aligned}$$

我们需定义 e, w, ϕ 使得 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, I_0(\sqsubseteq))$ 为良基集合且推理规则中的三个前提条件成立。

令 ϕ 如(b)中所定义。定义

$$\begin{aligned} w &= x \geq 0 \\ e &= h'(a, z, y_2) \end{aligned}$$

由 w 的定义知 $W = \{\sigma(x) \mid I(w)(\sigma) = true\} = \{0, 1, 2, \dots\}$ 为良基集合。

易知推理规则中的第一个条件和第二个条件成立。通过计算 $[T](\psi \vee (\phi \wedge e \sqsubseteq v))$ 知第三个条件亦成立，因而所需证明的结论成立。

前面 h' 的定义亦可换成如下定义。

$$\begin{aligned} I(h'(s_0, z, y_2)) &= 3z + 4 \\ I(h'(s_1, z, y_2)) &= 3(z + 1 - y_2 + y_1) \\ I(h'(s_2, z, y_2)) &= 3(z + 1 - y_2 + y_1) - 1 \\ I(h'(s_3, z, y_2)) &= 3(z + 1 - y_2 + y_1) + 1 \\ I(h'(s_4, z, y_2)) &= 0 \end{aligned}$$

例 7.4 以下讨论基于流程图模型语义的证明方法的应用例子。设 B, V 以及 T 和 I 如例 3.2 所定义。 $M = T$ 为 (B, V) 上的流程图模型。

(1) 证明 $\models_I \{z \geq 0\} T \{r = fib(z)\}$ 。

假设模型的运行路径如下。

$$(beg, \sigma_0)(s_1, \sigma_1)(s_2, \sigma_2)(s_3, \sigma_3)(s_1, \sigma_4)(s_2, \sigma_5) \cdots (s_1, \sigma_{n-1})(end, \sigma_n) \cdots$$

记路径上的状态的标号序列为 l_0, l_1, \dots, l_n 其中 $l_n = end$ 。

运行路径上的状态 (l_k, σ_k) 满足以下性质：

若 $l_k = s_1$ ，则

$$\begin{aligned} \sigma_k(z) &= \sigma_0(z) \wedge \\ \sigma_k(r) &\geq 1 \wedge \\ \sigma_k(y_1) &= fib(\sigma_k(r)) \wedge \\ \sigma_k(y_2) &= fib(\sigma_k(r) + 1) \wedge \\ \sigma_k(y_1) &\leq \sigma_k(z) \end{aligned}$$

以下用归纳法证明该性质。

(a) 对 $l \in \{l_0, l_2, l_3\}$ ，性质显然成立。根据语义，我们有

$$\begin{aligned} \sigma_1(z) &= \sigma_0(z) \wedge \\ \sigma_1(r) &= 1 \wedge \\ \sigma_1(y_1) &= 1 = fib(1) \wedge \\ \sigma_1(y_2) &= 1 = fib(1 + 1) \wedge \\ \sigma_1(y_1) &= 1 \leq \sigma_0(z) = \sigma_1(z) \end{aligned}$$

因而性质对 $l = l_1$ 成立。

(b) 以下证明对 $k \geq 4$ ，若性质对所有 $i < k$ 成立，则性质对 $i = k$ 成立。

若 $l_k \neq s_1$ ，性质显然成立。

以下假设 $l_k = s_1$ 。

我们有

$$\begin{aligned} (s_1, \sigma_{k-3}) &\rightarrow (s_2, \sigma_{k-2}) \\ (s_2, \sigma_{k-2}) &\rightarrow (s_3, \sigma_{k-1}) \\ (s_3, \sigma_{k-1}) &\rightarrow (s_1, \sigma_k) \end{aligned}$$

根据语义，我们有

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

根据 σ_k 与 σ_{k-3} 的关系以及性质在 σ_{k-3} 成立，可以得到

$$\sigma_k(z) = \sigma_0(z) \wedge \sigma_k(r) \geq 1 \wedge \sigma_k(y_1) = fib(\sigma_k(r)) \wedge \sigma_k(y_2) = fib(\sigma_k(r) + 1) \wedge \sigma_k(y_1) \leq \sigma_k(z)$$

(c) 根据归纳法，性质成立。

由模型的运行路径知, $l_{n-1} = s_1$ 且

$$\sigma_{n-1}(z) = \sigma_0(z) \wedge \sigma_{n-1}(r) \geq 1 \wedge \sigma_{n-1}(y_1) = fib(\sigma_{n-1}(r)) \wedge \sigma_{n-1}(y_2) = fib(\sigma_{n-1}(r) + 1) \wedge \sigma_{n-1}(y_1) \leq \sigma_{n-1}(z)$$

由于我们有 $(l_{n-1}, \sigma_{n-1}) \rightarrow (l_n, \sigma_n)$, 根据语义知 $\sigma_n(y_2) > \sigma_n(z)$ 且

$$\sigma_n(z) = \sigma_0(z) \wedge \sigma_n(y_1) = fib(\sigma_n(r)) \wedge \sigma_n(y_2) = fib(\sigma_n(r) + 1) \wedge \sigma_n(y_1) \leq \sigma_n(z)$$

因而有 $fib(\sigma_n(r)) \leq \sigma_n(z) \wedge \sigma_n(z) < fib(\sigma_n(r) + 1)$, 即 $\sigma_n(r) = bif(\sigma_n(z))$ 。

(2) 证明 $\models_I [z \geq 0]T[true]$ 。

假设模型的运行不终止且运行路径如下。

$$(beg, \sigma_0)(s_1, \sigma_1)(s_2, \sigma_2)(s_3, \sigma_3)(s_1, \sigma_4)(s_2, \sigma_5)(s_3, \sigma_6)(s_1, \sigma_7) \cdots$$

记 σ_k 对应的标号为 l_k 。

我们证明必然有一个 k 使得 $l_k = s_1$ 且 $I(y_2 \leq z)(\sigma_k)$ 不成立, 因而与模型不终止矛盾。

运行路径上的状态 (l_k, σ_k) 满足以下性质 (用归纳法可证):

$$l_{3i+1} = s_1 \text{ 且 } \sigma_{3i+1}(z) = \sigma_0(z) \wedge \sigma_{3i+1}(y_2) \geq i$$

选取 $k = 3(\sigma_0(z) + 1) + 1$, 则 $\sigma_k(y_2) = \sigma_0(z) + 1$ 且 $\sigma_k(z) = \sigma_0(z)$ 。因而 $I(y_2 \leq z)(\sigma_k) = \sigma_k(y_2) \leq \sigma_k(z)$ 不成立。

(3) 证明 $\models_I [z > 0]T[r = bif(x)]$ 。

假设模型的运行路径如下。

$$(beg, \sigma_0)(s_1, \sigma_1)(s_2, \sigma_2)(s_3, \sigma_3)(s_1, \sigma_4)(s_2, \sigma_5)(s_3, \sigma_6)(s_1, \sigma_7) \cdots$$

记 σ_k 对应的标号为 l_k 。

运行路径上的状态 (l_k, σ_k) 满足以下性质 (用归纳法可证):

若 $k < bif(z)$, 则 $(l_0, \sigma_0) \xrightarrow{*} (l_{3k+1}, \sigma_{3k+1})$ 且以下成立。

$$\begin{aligned} l_{3k+1} &= s_1 \\ \sigma_{3k+1}(z) &= \sigma_0(z) \\ \sigma_{3k+1}(r) &= k + 1 \\ \sigma_{3k+1}(y_1) &= fib(k + 1) \\ \sigma_{3k+1}(y_2) &= fib(k + 2) \end{aligned}$$

设 $k = bif(\sigma_0(z)) - 1$ 。

则 $\sigma_{3k+1}(y_2) = fib(k + 2) > \sigma_0(z) = \sigma_{3k+1}(z)$ 。

因而

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

根据语义知 $\sigma_{3k+2}(r) = \sigma_{3k+1}(r) = k + 1 = bif(\sigma_0(z))$ 。

例 7.5 设 B, V 以及 T 和 I 如例 3.2 所定义。 $M = T$ 为 (B, V) 上的流程图模型。

(1) (s_1, s_2, s_3, s_1) 与 $(y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z)$ 的最弱宽松前断言计算如下。

$$\begin{aligned} & \overline{wlp((s_1, s_2, s_3, s_1), y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z) =} \\ & [s_1, s_2, s_3, s_1](y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r+1) \leq z) = \\ & [s_1, s_2, s_3](y_1 = fib(r+1) \wedge y_2 = fib(r+1+1) \wedge fib(r+1) \leq z) = \\ & [s_1, s_2](y_2 = fib(r+1) \wedge y_1 + y_2 = fib(r+1+1) \wedge fib(r+1) \leq z) = \\ & (y_2 \leq z \rightarrow (y_2 = fib(r+1) \wedge y_1 + y_2 = fib(r+2) \wedge fib(r+1) \leq z)) \end{aligned}$$

(2) $\models_I \{y_1 = fib(r) \wedge y_2 = fib(r+1)\}(s_1, s_2, s_3, s_1)\{(y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z)\}$ 的基于验证条件计算的证明如下。

$\models_I \{y_1 = fib(r) \wedge y_2 = fib(r+1)\}(s_1, s_2, s_3, s_1)\{(y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z)\}$	iff
$vc(y_1 = fib(r) \wedge y_2 = fib(r+1), (s_1, s_2, s_3, s_1), (y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z))$	iff
$y_1 = fib(r) \wedge y_2 = fib(r+1) \rightarrow (s_1, s_2, s_3, s_1)[(y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z)]$	iff
$y_1 = fib(r) \wedge y_2 = fib(r+1) \rightarrow (y_2 \leq z \rightarrow (y_2 = fib(r+1) \wedge y_1 + y_2 = fib(r+2) \wedge fib(r+1) \leq z))$	iff
<i>true</i>	

例 7.6 设 B, V 以及 T 和 I 如例3.2所定义。 $M = T$ 为 (B, V) 上的流程图模型。

(1) 应用基于归纳断言的部分正确性证明方法证明 $\models_I \{z = c \wedge c > 0\}T\{r = bif(c)\}$ 。

以下列举证明的主要步骤。

1. 选取 $C = \{beg, s_1, end\}$ 。

2. 选取 $q_{beg}, q_{end}, q_{s_1}$ 。

$$\begin{aligned} q_{beg} & z > 0 \\ q_{end} & r = bif(z) \\ q_{s_1} & (y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z) \wedge r \geq 1 \end{aligned}$$

3. 列举相关路径

$$\begin{aligned} & (beg, s_1) \\ & (s_1, s_2, s_3, s_1) \\ & (s_1, s_4, end) \end{aligned}$$

4. 证明相关验证条件

$$\begin{aligned} & vc(q_{beg}, (beg, s_1), q_{s_1}) \\ & vc(q_{s_1}, (s_1, s_2, s_3, s_1), q_{s_1}) \\ & vc(q_{s_1}, (s_1, s_4, end), q_{end}) \end{aligned}$$

(2) 应用基于谓词的终止性证明方法证明 $\models_I [z > 0]T[true]$ 。

以下列举证明的主要步骤。

1. 选取 $C = \{beg, s_1\}$ 。

2. 选取 q_{beg}, q_{s_1}

$$\begin{aligned} q_{beg} & true \\ q_{s_1} & (y_1 = fib(r) \wedge y_2 = fib(r+1) \wedge fib(r) \leq z) \wedge r \geq 1 \end{aligned}$$

3. 列举相关路径

$$(beg, s_1) \\ (s_1, s_2, s_3, s_1)$$

4. 证明相关验证条件

$$vc(q_{beg}, (beg, s_1), q_{s_1}) \\ vc(q_{s_1}, (s_1, s_2, s_3, s_1), q_{s_1})$$

5. 选取 $C' = \{test\}$ 。

6. 选取 $(W, \sqsubseteq) = (\{0, 1, 2, \dots\}, \leq)$

7. 选取 $g_{s_1} : \Sigma \rightarrow W$

$$g_{s_1}(\sigma) = \sigma(z) + 1 - \sigma(y_2)$$

8. 列举相关路径(II)

$$(s_1, s_2, s_3, s_1)$$

9. 证明相关验证条件(II)

$$I(q_{s_1})(\sigma) = true \wedge (\sigma \xrightarrow{(s_1, s_2, s_3, s_1)} \sigma') \rightarrow g_{s_1}(\sigma') < g_{s_1}(\sigma)$$

(3) 应用基于公式的终止性证明方法证明 $\models_I [z > 0]T[true]$ 。

以下列举证明的主要步骤。

1. 选取 $C = \{beg, s_1\}$ 。

2. 选取 q_{beg}, q_{s_1}

$$q_{beg} \quad z > 0 \\ q_{s_1} \quad (y_1 = fib(r) \wedge y_2 = fib(r + 1) \wedge fib(r) \leq z) \wedge r \geq 1$$

3. 列举相关路径

$$(beg, s_1) \\ (s_1, s_2, s_3, s_1)$$

4. 证明相关验证条件

$$vc(q_{beg}, (beg, s_1), q_{s_1}) \\ vc(q_{s_1}, (s_1, s_2, s_3, s_1), q_{s_1})$$

5. 选取 $C' = \{s_1\}$ 。

6. 选取 $w = true$ 并证明 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, \leq)$ 为良基集合。

7. 选取 $t_{s_1} = z + 1 - y_2$ 并证明 $q_{s_1} \rightarrow w_x^{t_{s_1}}$

8. 列举相关路径(II)

$$(s_1, s_2, s_3, s_1)$$

9. 证明相关验证条件(II)

$$vc(q_{s_1} \wedge t_{s_1} = v, (s_1, s_2, s_3, s_1), t_{s_1} < v)$$

(4) 若将模型的解释改在整数上, 则对(2)的证明有以下影响。

对于证明中第7部分, 为保证 g_{s_1} 的值域在 W 中, 修改 g_{s_1} 如下。

$$g_{s_1}(\sigma) = \begin{cases} \sigma(z) + 1 - \sigma(y_2) & \text{若 } \sigma(y_2) \leq \sigma(z) \\ 0 & \text{否则。} \end{cases}$$

(5) 若将模型的解释改在整数上, 则对(3)的证明有以下影响。

对于证明中的第6部分, 选取 $w = (x \geq 0)$ 并证明 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, \leq)$ 为良基集合。

对于证明中的第7部分, 选取 $t_{s_1} = z + 1 - y_2 + y_1$ 并证明 $q_{s_1} \rightarrow w_x^{t_{s_1}}$

例 7.7 设 (B, V) 如例3.3所定义。定义 S 为以下字符串。

```

y1 := 1; y2 := 1; r := 1;
while (y2 ≤ z) do y0 := y1; y1 := y2; y2 := y0 + y1; r := r + 1 od;
∈

```

设 B 的解释 $I = (D, I_0)$ 如例3.2所定义。

定义 φ 如下。

$$\varphi = (y_1 = fib(r) \wedge y_2 = fib(r + 1) \wedge fib(r) \leq z) \wedge r \geq 1$$

记 T_0 为 $y_1 := 1; y_2 := 1; r := 1$, T_1 为 $y_0 := y_1; y_1 := y_2; y_2 := y_0 + y_1; r := r + 1$ 。

(1a) T_0 与 φ 的最弱宽松前断言 $wlp(T_0, \varphi)$ 的计算与化简如下:

$$\begin{aligned}
wlp(T_0, \varphi) &= \\
[T_0]\varphi &= \\
[y_1 := 1; y_2 := 1; r := 1]\varphi &= \\
[y_1 := 1; y_2 := 1]\varphi(r/1) &= \\
[y_1 := 1]\varphi(r/1)(y_2/1) &= \\
\varphi(r/1)(y_2/1)(y_1/1) &= \\
(1 = fib(1) \wedge 1 = fib(1 + 1) \wedge fib(1) \leq z) \wedge 1 \geq 1 &= \\
1 \leq z &=
\end{aligned}$$

(1b) T_1 与 φ 的最弱宽松前断言 $wlp(T_1, \varphi)$ 的计算如下:

$$\begin{aligned}
wlp(T_1, \varphi) &= \\
[T_1]\varphi &= \\
[y_0 := y_1; y_1 := y_2; y_2 := y_0 + y_1]\varphi(r/(r + 1)) &= \\
[y_0 := y_1; y_1 := y_2]\varphi(r/(r + 1))(y_2/(y_0 + y_1)) &= \\
[y_0 := y_1]\varphi(r/(r + 1))(y_2/(y_0 + y_1))(y_1/y_2) &= \\
\varphi(r/(r + 1))(y_2/(y_0 + y_1))(y_1/y_2)(y_0/y_1) &= \\
((y_1 = fib(r + 1) \wedge y_2 = fib(r + 2) \wedge fib(r + 1) \leq z) \wedge r + 1 \geq 1)(y_2/(y_0 + y_1))(y_1/y_2)(y_0/y_1) &= \\
((y_1 = fib(r + 1) \wedge y_0 + y_1 = fib(r + 2) \wedge fib(r + 1) \leq z) \wedge r + 1 \geq 1)(y_1/y_2)(y_0/y_1) &= \\
((y_2 = fib(r + 1) \wedge y_0 + y_2 = fib(r + 2) \wedge fib(r + 1) \leq z) \wedge r + 1 \geq 1)(y_0/y_1) &= \\
((y_2 = fib(r + 1) \wedge y_1 + y_2 = fib(r + 2) \wedge fib(r + 1) \leq z) \wedge r + 1 \geq 1) &=
\end{aligned}$$

(2a) $\models_I \{z > 0\}T_0\{\varphi\}$ 的证明如下。

$$\begin{aligned} \models_I \{z > 0\}T_0\{\varphi\} &\Leftrightarrow \\ \models_I z > 0 \rightarrow [T_0]\varphi &\Leftrightarrow \\ \models_I z > 0 \rightarrow 1 \leq z &\Leftrightarrow \\ \text{true} & \end{aligned}$$

(2b) $\models_I \{(y_2 \leq z) \wedge \varphi\}T_1\{\varphi\}$ 的证明如下。

$$\begin{aligned} \models_I \{(y_2 \leq z) \wedge \varphi\}T_1\{\varphi\} &\Leftrightarrow \\ \models_I (y_2 \leq z) \wedge \varphi \rightarrow [T_1]\varphi &\Leftrightarrow \\ \models_I (y_2 \leq z) \wedge \varphi \rightarrow (y_2 = \text{fib}(r+1) \wedge y_1 + y_2 = \text{fib}(r+2) \wedge \text{fib}(r+1) \leq z) \wedge r+1 \geq 1 &\Leftrightarrow \\ \text{true} & \end{aligned}$$

例 7.8 设 (B, V) , S , I 如例7.7所定义。 $\models_I \{z > 0\}S\{r = \text{bif}(z)\}$ 的证明如下。

设 φ 为如下公式。

$$\varphi = (y_1 = \text{fib}(r) \wedge y_2 = \text{fib}(r+1) \wedge \text{fib}(r) \leq z) \wedge r \geq 1$$

由例7.7知 $\models_I \{(y_2 \leq z) \wedge \varphi\}T_1\{\varphi\}$ 。

由循环规则知 $\models_I \{\varphi\}\text{while } (y_2 \leq z) \text{ do } T_1 \text{ od } \{\neg(y_2 \leq z) \wedge \varphi\}$ 。

我们有 $\neg(y_2 \leq z) \wedge \varphi \rightarrow r = \text{bif}(x)$ 。

由推论规则知 $\models_I \{\varphi\}\text{while } (y_2 \leq z) \text{ do } T_1 \text{ od } \{r = \text{bif}(z)\}$ 。

由例7.7知 $\models_I \{z > 0\}T_0\{\varphi\}$ 。

由组合规则知 $\models_I \{z > 0\}S\{r = \text{bif}(z)\}$ 。

例 7.9 设 (B, V) , S , I 如例7.7所定义。

(1) 设 I 为 B 在自然数上的解释如例7.7所定义。 $\models_I [z > 0]S[r = \text{bif}(x)]$ 的证明如下。

设 w, t, φ 如下。

$$\begin{aligned} w &= (\text{true}) \\ t &= (z+1 - y_2) \\ \varphi &= (y_1 = \text{fib}(r) \wedge y_2 = \text{fib}(r+1) \wedge \text{fib}(r) \leq z) \wedge r \geq 1 \end{aligned}$$

由 w 的定义知 $W = \{\sigma(x) \mid I(w)(\sigma) = \text{true}\} = \{0, 1, 2, \dots\}$ 为良基集合。

我们有 $(y_2 \leq z) \wedge \varphi \rightarrow w(x/t)$ 。

类似于7.7中的相关证明可得 $\models_I [\varphi \wedge (y_2 \leq z \wedge t = v)]T_1[\varphi \wedge t < v]$ 。

由循环规则知 $\models_I [\varphi]\text{while } (y_2 \leq z) \text{ do } T_1 \text{ od } \{\neg(y_2 \leq z) \wedge \varphi\}$ 。

我们有 $\neg(y_2 \leq z) \wedge \varphi \rightarrow r = \text{bif}(x)$ 。

由推论规则知 $\models_I [\varphi]\text{while } (y_2 \leq z) \text{ do } T_1 \text{ od } \{r = \text{bif}(x)\}$ 。

类似于7.7中的相关证明可得 $\models_I [z > 0]T_0[\varphi]$ 。

由组合规则知 $\models_I [z > 0]S[r = \text{bif}(x)]$ 。

(2) 设 I 为 B 在整数上的解释, 其余各项类似于7.7中的定义。

修改 w 为 $x \geq 0$ 。则 $W = \{\sigma(x) \mid I(w)(\sigma) = \text{true}\} = \{0, 1, 2, \dots\}$ 为良基集合。证明的其它部分类似。

§7.5 练习

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

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

给定 I 为 B 在整数上的正常解释。

- (1) 计算最弱宽松前断言 $wlp(T, a = s_4)$ 即 $[T](a = s_4)$ 。
 (2) 证明 $(T, \Theta) \models (a = s_3) \rightarrow O(a = s_4)$ 。
2. 设 $(B, V), (T, \Theta), I$ 如练习1所定义。
 (1) 证明 $(T, \Theta) \models_I n \geq 0 \rightarrow \Box(a = s_4 \rightarrow y = n * n * n - n)$;
 (2) 证明 $(T, \Theta) \models_I n \geq 0 \rightarrow \Diamond(a = s_4)$ 。
3. 设 $B = (\{0, 1, 2, 3, +, -, *\}, \{<, =, >\})$ 和 $V = \{n, x, y\}$ 。给定以下模型 T 。

$$\begin{aligned} beg & : (x, y) := (0, 0) \text{ goto } l_1 \\ l_1 & : \text{ if } (x < n) \text{ goto } l_2 \text{ else goto } l_3 \\ l_2 & : (x, y) := (x + 1, y + x * (x + 1)) \text{ goto } l_1 \\ l_3 & : (y) := (3 * y) \text{ goto } end \end{aligned}$$

给定 I 为 B 在整数上的正常解释。

- (1) 应用语义证明 $\models \{(x \leq n) \wedge 3 * y = x * x * x - x\}(l_1, l_3, end)\{y = n * n * n - n\}$ 。
 (2) 计算 $wlp(l_1, l_3, end), (y = n * n * n - n)$ 并应用验证条件证明 (1) 的路径部分正确性。
4. 设 $(B, V), T, I$ 如练习3所定义。
 (1) 证明 T 对于前断言 $n \geq 0$ 和后断言 $y = n * n * n - n$ 部分正确。
 (2) 证明 T 对于前断言 $n \geq 0$ 能够终止。
5. 设 $B = (\{1, 10, 100, +, -\}, \{<, =, >\})$ 和 $V = \{x, y_1, y_2, z\}$ 。给定以下模型 T 。

$$\begin{aligned} beg & : (y_1, y_2) := (x, 1) \text{ goto } test_1 \\ test_1 & : \text{ if } (y_1 > 100) \text{ goto } test_2 \text{ else goto } update_1 \\ test_2 & : \text{ if } (y_2 = 1) \text{ goto } res \text{ else goto } update_2 \\ update_1 & : (y_1, y_2) := (y_1 + 11, y_2 + 1) \text{ goto } test_1 \\ update_2 & : (y_1, y_2) := (y_1 - 10, y_2 - 1) \text{ goto } test_1 \\ res & : (z) := (y_1 - 10) \text{ goto } end \end{aligned}$$

给定 I 为 B 在自然数上的正常解释。

(1) 证明 T 对于前断言 $x \leq 100$ 和后断言 $z = 91$ 部分正确。

(2) 证明 T 对于前断言 $x \geq 0$ 能够终止。

6. 设 $B = (\{0, 1, 2, 3, +, -, *\}, \{<, =, >\})$ 和 $V = \{i, j, k, l, x, y, a, b\}$ 。给定以下程序片段 T_1 。

```
if (x > y)
then x := x - y; i := i - k; j := j - l;
else y := y - x; k := k - i; l := l - j;
fi
```

给定 I 为 B 在整数上的正常解释。

(1) 计算 $wlp(T_1, (x = i * a + j * b))$ 并化简。

(2) 证明 $\models_I \{(y = k * a + l * b) \wedge (x = i * a + j * b)\} T_1 \{x = i * a + j * b\}$ 。

7. 设 $(B, V), I, T_1$ 如练习6所定义。

给定以下程序 S : $i := 1; j := 0; k := 0; l := 0; \text{while } \neg(x = y) \text{ do } T_1 \text{ od}; \epsilon$

(1) 证明 $\models_I \{x = a \wedge y = b \wedge a \geq 0 \wedge b \geq 0\} S \{x = \text{gcd}(a, b) \wedge x = i * a + j * b\}$ 。

(2) 证明 $\models_I [x = a \wedge y = b \wedge a > 0 \wedge b > 0] S [x = \text{gcd}(a, b) \wedge x = i * a + j * b]$ 。

§8 基于模型检测的验证方法

模型检测(Model Checking)的含义是检查一个模型是否满足给定的性质,特别是使用计算的方法检查一个模型是否满足给定的性质。模型检测方法可分为三类:基于状态的分析,基于符号模型的分析,基于时序逻辑限界语义的分析。基于状态的分析是一类基本的分析方法,通过查看状态及状态之间的关系判定给定的性质是否成立。基于符号模型的分析是将状态集合用命题逻辑公式(布尔公式)表示,通过对公式的计算来判定给定的性质是否成立。基于时序逻辑限界语义的分析主要是结合时序逻辑公式的限界语义与符号模型,把模型检测问题转换为命题逻辑公式的可满足性问题。限界正确性检查的主要思想是对给定性质,在给定的限制下,试图证明性质的正确且同时试图证明性质的不正确,因而若两者之一有肯定的答案,则知道模型是否满足性质。限界模型检测的主要思想是对给定性质,将其取反,在给定的限制下,试图证明模型有初始状态满足取反后的性质,由此得出模型不满足给定性质。

§8.1 符号模型

本节论述符号模型,介绍基于符号模型的分析的基础知识。

设 V 为命题集合。定义 $V' = \{v' \mid v \in V\}$ 。记 $BF(V)$ 为命题集合 V 上的布尔公式的集合。记 φ' 为将 φ 中每个变量 x 用 x' 替换后的公式。

定义 8.1 给定一个命题集合 AP 。一个 AP 上的符号模型是一个四元组 $\langle V, \rho, \Theta, N \rangle$ 其中 V 为命题变量集合, $\rho \in BF(V \cup V')$ 为 $V \cup V'$ 上的布尔公式, $\Theta \in BF(V)$ 为 V 上的布尔公式, $N : AP \rightarrow BF(V)$ 为 AP 到 V 上公式集合 $BF(V)$ 的映射。

设 $V = \{v_1, \dots, v_n\}$ 。

记 $v = (v_1, \dots, v_n)$ 。若 $s = (a_1, \dots, a_n)$, 则 q_v^a 表示 $q_{v_1, \dots, v_n}^{a_1, \dots, a_n}$ 。

记 $q_v^a = 1$ 为 $s \models q$ 。

系统状态: 系统状态由命题变量 v_1, \dots, v_n 决定。一个系统状态,即 v_1, \dots, v_n 的取值,用 n 元组 (a_1, \dots, a_n) 表示。系统状态的集合为 $\{0, 1\}^n$ 。

系统的迁移关系 R_ρ 为 $\{(a_1, \dots, a_n), (a'_1, \dots, a'_n) \mid (a_1, \dots, a_n, a'_1, \dots, a'_n) \models \rho\}$ 。

系统的初始状态集合 I_Θ 为 $\{(a_1, \dots, a_n) \mid (a_1, \dots, a_n) \models \Theta\}$ 。

路径和计算: 我们用 $s \rightarrow s'$ 表示存在从 s 到 s' 的迁移,即 $(s, s') \in R_\rho$ 。符号模型 $M = \langle V, \rho, \Theta, N \rangle$ 上的一条路径是状态集 $\{0, 1\}^n$ 上的一个满足对任意 $i \geq 0$ 有 $s_i \rightarrow s_{i+1}$ 的无穷序列 $[s_i]_{i \geq 0}$ 。符号模型 M 上的一次计算是该符号模型上的第一个状态满足 Θ 的一条路径。

符号模型的标号Kripke结构: 给定 AP 上的符号模型 $M = \langle V, \rho, \Theta, N \rangle$ 。

定义 $S = \{0, 1\}^n$ 。

定义 $R = \{(s, s') \mid \rho_{v, v'}^{s, s'} = true\}$ 。

定义 $I = \{s \mid \Theta_v^s = true\}$ 。

定义 $L(s) = \{p \in AP \mid s \models N(p)\}$ 。

则我们有与 M 对应的 AP 上的Kripke结构 $K_M = \langle S, R, I, L \rangle$ 。

定义 8.2 设 φ 为 AP 上的CTL或PLTL公式。 $M \models \varphi$ 当且仅当 $K_M \models \varphi$ 。

符号模型的特殊类型： 设 $M = \langle V, \rho, \Theta, N \rangle$ 为 AP 上的符号模型。若 $AP = V$ 且对对 $p \in AP$ 有 $N(p) = p$ ，则 M 可简写为 $\langle V, \rho, \Theta \rangle$ 。

标号 Kripke 结构的符号模型： 给定 AP 上的标号 Kripke 结构 $M = \langle S, R, I, L \rangle$ 。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 v_1, \dots, v_n 来表示。 S 中的第 i 个元素记为 $s(i-1)$ 。

设 $f_0 : S \rightarrow \{0, 1\}^n$ 为单射函数。

设 $f_0(s) = (a_1, \dots, a_n)$ 。

记 $f_0(s, 0) = \{v_i \mid a_i = 0, i = 1, \dots, n\}$ 和 $f_0(s, 1) = \{v_i \mid a_i = 1, i = 1, \dots, n\}$ 。

定义 $f_1(s) = \bigwedge_{v \in f_0(s, 1)} v \wedge \bigwedge_{v \in f_0(s, 0)} \neg v$ 。

定义 $\Theta = \bigvee_{s \in I} f_1(s)$ 。

定义 $f_2(s, s') = f_1(s) \wedge (f_1(s'))'$ 。

定义 $\rho = \bigvee_{(s, s') \in R} f_2(s, s')$ 。

对每个 $p \in AP$ ，定义 $N(p) = \bigvee_{p \in L(s)} f_1(s)$ 。

定义 $V = \{v_1, \dots, v_n\}$ 。

则我们有与 $M = \langle S, R, I, L \rangle$ 对应的符号模型 $(M)^{sm} = \langle V, \rho, \Theta, N \rangle$ 。

命题 8.1 设 M 为 AP 上的标号 Kripke 模型 M ， φ 为 AP 上的 CTL 或 PLTL 公式。 $M \models \varphi$ 当且仅当 $(M)^{sm} \models \varphi$ 。

前驱状态集 给定标号 Kripke 模型 $K = \langle S, R, I, L \rangle$ 和相应的基于 f_0 的符号模型 $M = \langle V, \rho, \Theta, N \rangle$ 。设 $\varphi \in BF(V)$ 为 $V = \{x_1, \dots, x_n\}$ 上的布尔公式。

用 $s \models \varphi$ 表示 $f_0(s) \models \varphi$ 。

若 $s \models \varphi$ 则称 s 为 φ 状态。

用 φ' 表示 $\varphi_{x_1, \dots, x_n}^{x'_1, \dots, x'_n}$ 。

定义 $\exists x. \varphi = (\varphi_x^0 \vee \varphi_x^1)$ 。定义函数 $ex : BF(V) \rightarrow BF(V)$ 如下：

$$ex(\varphi) = \exists x'_1 \dots x'_n. (\rho(x_1, \dots, x_n, x'_1, \dots, x'_n) \wedge \varphi')$$

则 $ex(\varphi)$ 表示 φ 状态的相应前驱状态的集合。

命题 8.2 $\{s \mid s \models ex(\varphi)\} = \{s \mid s \rightarrow s', s' \models \varphi\}$ 。

后继状态集 设 $\psi \in BF(V')$ 为 $V' = \{x'_1, \dots, x'_n\}$ 上的布尔公式。

用 $\psi^\#$ 表示 $\psi_{x'_1, \dots, x'_n}^{x_1, \dots, x_n}$ 。

定义函数 $xe : BF(V) \rightarrow BF(V)$ 如下：

$$xe(\varphi) = (\exists x_1 \dots x_n. (\rho(x_1, \dots, x_n, x'_1, \dots, x'_n) \wedge \varphi))^\#$$

则 $xe(\varphi)$ 表示 φ 状态的相应后继状态的集合。

命题 8.3 $\{s \mid s \models xe(\varphi)\} = \{s \mid s' \rightarrow s, s' \models \varphi\}$ 。

定义 8.3 给定一个命题集合 AP 。一个 AP 上的公平符号模型是一个五元组 $\langle V, \rho, \Theta, N, F \rangle$ 其中 $\langle V, \rho, \Theta, N \rangle$ 为符号模型， F 为 V 上的公式的集合。

给定 $M = \langle V, \rho, \Theta, N, F \rangle$ 。

系统状态即是模型 $M' = \langle V, \rho, \Theta, N \rangle$ 的状态。系统的路径和计算即是 M' 的路径和计算。

公平路径和公平计算： 无穷序列 $\pi = [s_i]_{i \geq 0}$ 是系统的公平路径，当且仅当 π 是对每一个 $f \in F$ 有无穷多 s_i 满足 f 的路径。无穷序列 $\pi = [s_i]_{i \geq 0}$ 是系统的计算，当且仅当 s_0 满足 Θ 且 π 是系统的公平路径。

公平状态： 状态 s 是公平的，当且仅当有一条起点是 s 的公平路径。

命题 8.4 定义 $fair = \nu Z. \bigwedge_{f \in F} \mu Y. ((f \wedge ex(Z)) \vee ex(Y))$ 。 s 是公平状态当且仅当 s 满足 $fair$ 。

公平符号模型的交： 设 $M_1 = \langle V_1, \rho_1, \Theta_1, N_1, F_1 \rangle$ 和 $M_2 = \langle V_2, \rho_2, \Theta_2, N_2, F_2 \rangle$ 是 AP 上的公平符号模型且 $V_1 \cap V_2 = \emptyset$ 。

定义 $\rho = \rho_1 \wedge \rho_2 \wedge \bigwedge_{p \in AP} (N_1(p)' \leftrightarrow N_2(p)')$ 。

定义 $\Theta = \Theta_1 \wedge \Theta_2 \wedge \bigwedge_{p \in AP} (N_1(p) \leftrightarrow N_2(p))$ 。

对每个 $p \in AP$ ，定义 $N(p) = N_1(p) \wedge N_2(p)$ 。

则 $M_1 \cap M_2 = \langle V_1 \cup V_2, \rho, \Theta, N, F_1 \cup F_2 \rangle$ 是公平符号模型。

命题 8.5 若 π 是 M_1 的公平计算且 π' 是 M_2 的公平计算且 $\forall p \in AP. \forall i \geq 0. (\pi_i \models p \leftrightarrow \pi'_i \models p)$ ，则存在 $M_1 \cap M_2$ 的公平计算 π'' 使得 $\forall p \in AP. \forall i \geq 0. (\pi_i \models p \leftrightarrow \pi''_i \models p)$ ；若 π 是 $M_1 \cap M_2$ 的公平计算，则存在 M_1 的公平计算 π' 和 M_2 的公平计算 π'' 使得 $\forall p \in AP. \forall i \geq 0. ((\pi_i \models p \leftrightarrow \pi'_i \models p) \wedge (\pi_i \models p \leftrightarrow \pi''_i \models p))$ 。

符号模型与公平符号模型的交： 设 $M = \langle V, \rho, \Theta, N \rangle$ 为符号模型。则 M 可以看成是公平符号模型 $M' = \langle V, \rho, \Theta, N, \{\} \rangle$ 。符号模型与公平符号模型的交则可以看成是公平符号模型的交的特殊情况。

§8.2 CTL性质的模型检测

本节分三小节分别介绍基于状态的分析，基于符号模型的分析以及基于时序逻辑限界语义的分析。

§8.2.1 CTL性质的状态标号算法与不动点算法

考虑由 $\{\vee, \neg, EX, EG, EU\}$ 构成的 CTL 逻辑联接符和时序算子的完全集。

状态标号算法 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。

一个状态是否满足原子命题由 L 给定。

假设每个状态是否满足 φ 的子公式已经计算过。对于一个状态是否满足 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 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 。

命题 8.6 $M, s \models \varphi$ 当且仅当 $\varphi \in L(s)$ 。

推论 8.1 $M \models \varphi$ 当且仅当 $\varphi \in \bigcap_{s \in I} L(s)$ 。

不动点算法 给定 AP 上的 Kripke 结构 $\langle S, R, I, L \rangle$ 和一个 CTL 公式 φ , 该不动点算法的主要思想是对每个 φ 的子公式逐层计算哪些状态满足这些子公式, 直至最后知道哪些状态满足 φ 。 M 满足 φ 当且仅当 I 中的状态都满足 φ 。具体算法见 §6.2 节及推论 6.3。

§8.2.2 CTL 性质的符号模型检测

CTL 性质的符号模型检测建立在 CTL 性质的不动点算法之上, 用布尔公式表示状态集合, 通过对公式的计算来判定给定的性质是否成立。考虑由 $\{\vee, \neg, EX, EG, EU\}$ 构成的 CTL 逻辑联接符和时序算子的完全集。

公式的不动点: 设 $f : BF(V) \rightarrow BF(V)$ 。定义 $f^0(\varphi) = \varphi$ 和 $f^{i+1}(\varphi) = f^i(\varphi)$ 。 $f(Z)$ 的最小不动点, 记作 $\mu Z.f(Z)$, 定义为 $f^k(false)$, 其中 k 为最小的满足 $f^{i+1}(false) \equiv f^i(false)$ 的 i 。 $f(Z)$ 的最大不动点, 记作 $\nu Z.f(Z)$, 定义为 $f^k(true)$, 其中 k 为最小的满足 $f^{i+1}(true) \equiv f^i(true)$ 的 i 。

CTL 公式的不动点: 设 q 是 CTL 公式。我们分 6 种情况计算 q 成立的状态的集合对应的布尔公式。

(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 $[[q]] = \mu Z.([[q_1]] \vee ([[q_0]] \wedge ex(Z)))$

定义 8.4 $M \models \varphi$ 当且仅当 $\Theta \rightarrow [[\varphi]]$ 。

推论 8.2 给定标号 Kripke 模型 M 和 CTL 公式 φ 。设 $(M)^{sm} = \langle V, \rho, \Theta, N \rangle$ 。 $M \models \varphi$ 当且仅当 $\Theta \rightarrow [[\varphi]]$ 。

高效的布尔公式的表示与运算 为高效计算公式的且或非等运算以及检查一个公式是否为不动点, 通常用有序二元决策图(OBDD)表示布尔公式, 并在OBDD上进行相关运算(见§8.5节)。

§8.2.3 CTL性质的限界正确性检查

给定符号模型 $M = \langle V, \rho, \Theta, N \rangle$ 。给定CTL公式 φ 。

设 $v = \{v_1, \dots, v_n\}$ 。则一个状态变量 w 为一个 n 元组 (w_1, \dots, w_n) 。

设 $k \geq 0$ 。

用 \vec{u} 表示状态变量列表 u_0, \dots, u_k 。定义 $P_k(\vec{u})$ 如下。

$$P_k(\vec{u}) := \bigwedge_{j=0}^{k-1} T(u_j, u_{j+1})$$

定义 $\gamma_k(\vec{u})$ 如下。

$$\gamma_k(\vec{u}) := \bigvee_{x=0}^{k-1} \bigvee_{y=x+1}^k u_x = u_y.$$

定义 8.5 (CTL公式的转换) 设 $k \geq 0$ 。设 w 为状态变量。定义 $[[\varphi, w]]_k$ 如下。

$[[p, w]]_{k,l}$	$= (N(p))_v^w$
$[[\neg p, w]]_{k,l}$	$= \neg(N(p))_v^w$
$[[\varphi \vee \psi, w]]_k$	$= [[\varphi, w]]_k \vee [[\psi, w]]_k$
$[[\varphi \wedge \psi, w]]_k$	$= [[\varphi, w]]_k \wedge [[\psi, w]]_k$
$[[A\varphi, w]]_k$	$= \forall \vec{u}. (P(\vec{u}) \wedge w = u_0 \rightarrow [[\varphi, \vec{u}]]_k)$
$[[E\varphi, w]]_k$	$= \exists \vec{u}. (P(\vec{u}) \wedge w = u_0 \wedge [[\varphi, \vec{u}]]_k)$
$[[X\varphi, \vec{u}]]_k$	$= k \geq 1 \wedge [[\varphi, u_1]]_k$
$[[F\psi, \vec{u}]]_k$	$= \bigvee_{j=0}^k [[\psi, u_j]]_k$
$[[G\psi, \vec{u}]]_k$	$= \bigwedge_{j=0}^k [[\psi, u_j]]_k \wedge \gamma_k(\vec{u})$
$[[\varphi U \psi, \vec{u}]]_k$	$= \bigvee_{j=0}^k ([[\psi, u_j]]_k \wedge \bigwedge_{t=0}^{j-1} [[\varphi, u_t]]_k)$
$[[\varphi R \psi, \vec{u}]]_k$	$= \bigwedge_{j=0}^k ([[\psi, u_j]]_k \vee \bigvee_{t=0}^{j-1} [[\varphi, u_t]]_k) \wedge (\bigvee_{t=0}^k [[\varphi, u_t]]_k \vee \gamma_k(\vec{u}))$

定理 8.1 给定标号Kripke模型 M 和CTL公式 φ 。设 $(M)^{sm} = \langle V, \rho, \Theta, N \rangle$ 。 $M, s \models_k \varphi$ 当且仅当 $f_1(s) \rightarrow [[\varphi, v]]_k$ 成立。

推论 8.3 给定标号Kripke模型 M 和CTL公式 φ 。设 $(M)^{sm} = \langle V, \rho, \Theta, N \rangle$ 。 $M \models_k \varphi$ 当且仅当 $\Theta \rightarrow [[\varphi, v]]_k$ 成立。

限界正确性检查算法: 给定模型 M 和公式 φ 。限界正确性检查算法 $CTLbmc(M, \varphi)$ 如下:

- (1) $k = 0$;
- (2) 若 $\Theta \rightarrow [[\varphi, v]]_k$ 成立, 则return true;
- (3) 若 $\exists \vec{v}. (\Theta \wedge [[\neg \varphi, v]]_k)$ 成立, 则return false;
- (4) $k = k + 1$, 回到(2);

注意: 这里的 $\neg \varphi$ 代表与其等价的NNF范式的CTL公式。

推论 8.4 $M \models \varphi$ iff $CTLbmc(M, \varphi) = true$ 。

ACTL性质 若NNF范式公式中没有 E 出现, 则称这类性质为ACTL性质。对于这类性质的验证, 可根据限界语义将验证问题转化为(不带量词的)命题逻辑公式可满足性问题, 以提高验证效率。

§8.3 PLTL性质的模型检测

本节同样分三小节分别介绍基于状态的分析, 基于符号模型的分析以及基于时序逻辑限界语义的分析。

§8.3.1 PLTL性质的基于自动机空性检测的算法

给定 AP 上的Kripke结构 $M = \langle S, R, I, L \rangle$ 和一个PLTL公式 φ , 该算法的主要思想是将 $\neg\varphi$ 转换到等价的泛Büchi自动机 $GB_{\neg\varphi}$ (参考§5.1.3节), 将 M 转换到等价的泛Büchi自动机 GB_M (参考§4.2.3节), 根据命题4.16用空性检测算法检查 $GB_M \cap GB_{\neg\varphi}$ 是否为空。

命题 8.7 设 φ 为PLTL公式。 $M \models \varphi$ 当且仅当 $GB_M \cap GB_{\neg\varphi} \equiv \emptyset$ 。

为方便局部搜索(称为动态模型检测)以提高(主要是性质不满足时的)验证效率, 可将PLTL公式 $\neg\varphi$ 的泛Büchi自动机进一步转换到Büchi自动机 $B_{\neg\varphi}$, 将 M 转换到等价的Büchi自动机 B_M , 根据命题4.2用DDFS算法检查 $B_M \cap B_{\neg\varphi}$ 是否为空。

§8.3.2 PLTL性质的符号模型检测

由PLTL公式生成公平符号模型: 给定PLTL公式 φ 。

根据§8.3.1节的构造生成 S, I, F 。

根据§8.2.2节的原理定义 $V = \{v_1, \dots, v_n\}$ 以及 f_0, f_1, f_2 。

定义 $\Theta = \bigvee_{s \in I} f_1(s)$ 。

定义 $\rho = \bigvee_{s \in s'.a} f_2((s, s'))$ 。

对每个 $p \in AP$, 定义 $N(p) = \bigvee_{s.c \models p} f_1(s)$ 。

定义 $F' = \bigvee_{s \in F} f_1(s)$ 。

则 $(\varphi)^{fsm} = \langle V, \rho, \Theta, N, F' \rangle$ 是一个公平符号模型。

命题 8.8 给定标号Kripke结构 M 和PLTL公式 φ 。设 $(M)^{sm} \cap (\neg\varphi)^{fsm} = \langle V, \rho, \Theta, N, F \rangle$ 。则 $M \models \varphi$ 当且仅当 $\Theta \wedge fair$ 不可满足。

§8.3.3 PLTL公式的限界模型检测

给定符号模型 $M = \langle V, \rho, \Theta, N \rangle$ 。给定PLTL公式 φ 。

设 $v = \{v_1, \dots, v_n\}$ 。则一个状态变量 w 为一个 n 元组 (w_1, \dots, w_n) 。

设 $k \geq 0$ 。

用 $s(i, k, l)$ 表示若 $(k = i)$ 则 l 否则 $i + 1$ 。

定义 8.6 (PITL公式的转换) 设 $k \geq 0$ 。设 w 为状态变量。用 $[[\varphi, u_{-1}]]_{k,l}$ 表示false。定义 $[[\varphi, w]]_{k,l}$ 如下, 其中 $g(u, a, b, \varphi, \psi)$ 表示 $\bigvee_{j=a}^b ([[\psi, u_j]]_{k,l} \wedge \bigwedge_{t=a}^{j-1} [[\varphi, u_t]]_{k,l})$ 。

$[[p, w]]_{k,l}$	$= (N(p))_v^w$
$[[\neg p, w]]_{k,l}$	$= \neg(N(p))_v^w$
$[[\varphi \vee \psi, w]]_{k,l}$	$= [[\varphi, w]]_{k,l} \vee [[\psi, w]]_{k,l}$
$[[\varphi \wedge \psi, w]]_{k,l}$	$= [[\varphi, w]]_{k,l} \wedge [[\psi, w]]_{k,l}$
$[[X\varphi, u_i]]_{k,l}$	$= [[\varphi, u_{s(i,k,l)}]]_{k,l}$
$[[G\varphi, u_i]]_{k,l}$	$= \bigwedge_{j=i}^k [[\varphi, u_j]]_{k,l} \wedge \bigwedge_{j=\min(i,l)}^{i-1} [[\varphi, u_j]]_{k,l}$
$[[\varphi U\psi, u_i]]_{k,l}$	$= g(u, i, k, \varphi, \psi) \vee (\bigwedge_{t=i}^k [[\varphi, u_t]]_{k,l} \wedge g(u, l, i-1, \varphi, \psi))$

定义 8.7 $[[\varphi, \vec{u}]]_k := \bigvee_{l=-1}^k (T(u_k, u_l) \wedge [[\varphi, u_0]]_{k,l})$

定理 8.2 给定标号Kripke模型 M 和PLTL公式 φ 。设 $(M)^{sm} = \langle V, \rho, \Theta, N \rangle$ 。 $M, s \models_k^E \varphi$ 当且仅当 $(f_1(s))_v^{u_0} \wedge P_k(\vec{u}) \wedge [[\varphi, \vec{u}]]_k$ 可满足。

推论 8.5 给定标号Kripke模型 M 和PLTL公式 φ 。给定任意 $k \geq 0$ 。(1) 若 $\Theta_v^{u_0} \wedge P_k(\vec{u}) \wedge [[\neg\varphi, \vec{u}]]_k$ 可满足，则 $M \not\models \varphi$ ；(2) 若 $\Theta_v^{u_0} \wedge P_k(\vec{u}) \wedge [[\neg\varphi, \vec{u}]]_k$ 不可满足且 $k \geq lct((M)^{sm}, \varphi)$ ，则 $M \models \varphi$ 。

限界模型检测算法： 给定标号Kripke模型 M 和PLTL公式 φ 。设 $(M)^{sm} = \langle V, \rho, \Theta, N \rangle$ 。给定参数 b 。限界模型检测算法 $PLTLbmc(M, \varphi, b)$ 如下：

- (1) $k = 0$
- (2) 若 $\Theta_v^{u_0} \wedge P_k(\vec{u}) \wedge [[\neg\varphi, \vec{u}]]_k$ 可满足，则return false；
- (3) 若 $k = b$ ，则return unknown
- (4) $k = k + 1$ ，回到(2)

注意：这里的 $\neg\varphi$ 代表与其等价的NNF范式的PLTL公式。

推论 8.6 给定任意 b 。若 $PLTLbmc(M, \varphi, b) = false$ ，则 $M \not\models \varphi$ 。

推论 8.7 给定 $b \geq lct((M)^{sm}, \varphi)$ 。若 $PLTLbmc(M, \varphi, b) \neq false$ ，则 $M \models \varphi$ 。

§8.4 公平标号Kripke结构的模型检测

上两节的论述主要建立在标号Kripke结构模型上，本节论述建立在公平标号Kripke结构模型上的模型检测方法的一些基本思想。给定公平标号Kripke结构 $M = \langle S, R, I, L, F \rangle$ 且 $F = \{f_1, \dots, f_k\}$ 。

CTL性质 公平标号Kripke结构的CTL性质的基于不动点算法的显示状态模型检测，可依据§6.2.3节的计算方法进行。公平标号Kripke结构的CTL性质的符号模型检测，则首先将 M 表示为公平符号模型 $M' = \langle V, \rho, \Theta, N, \Phi \rangle$ 。用 $[[\varphi]]$ 表示满足 φ 的状态的集合的布尔公式。在§8.2.2节的基础上，我们有以下定义。

$fair$	$= \nu Z.(\bigwedge_{i=1}^k [[EX(EF(Z \wedge \phi_i))]])$
$[[EX_f p]]$	$= [[EX(p \wedge fair)]]$
$[[E(pU_f q)]]$	$= [[E(pU(q \wedge fair))]]$
$[[EG_f p]]$	$= \nu Z.([[p]] \wedge \bigwedge_{i=1}^k [[EX(E(pU(Z \wedge \phi_i))]])$

设 φ 为时序算子在 $\{EX_f, EU_f, EG_f\}$ 中的CTL性质。

命题 8.9 $M \models \varphi$ 当且仅当 $\Theta \rightarrow [[\varphi]]$ 。

LTL性质 公平标号Kripke结构的LTL性质的模型检测,可依据§5.2.3节的方法将其转化为去掉公平约束的标号Kripke结构的模型检测。对于基于自动机空性检测的算法,由于复杂公式转化成自动机的复杂度较高,亦可考虑将公平标号Kripke结构转换到等价的泛Büchi自动机 GB_M 且将 $\neg\varphi$ 转换到等价的泛Büchi自动机 $GB_{\neg\varphi}$ 。根据以下命题,用空性检测算法检查 $GB_M \cap GB_{\neg\varphi}$ 是否为空以验证 $M \models \varphi$ 是否成立。

命题 8.10 设 φ 为PLTL公式。 $M \models \varphi$ 当且仅当 $GB_M \cap GB_{\neg\varphi} \equiv \emptyset$ 。

§8.5 模型检测技术

本节介绍模型检测方法中提高模型检测效率的部分技术手段与优化策略。

§8.5.1 二元决策图

符号模型检测方法的基本思想是使用命题逻辑公式表示状态集合。模型检测的计算过程需要用到公式的且或非等运算。二元决策图是一种命题逻辑公式的范式表示方法且具有较为高效的且或非运算。

定义 8.8 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; n_0 为图的初始点; $E: N \rightarrow N \times N$ 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; $L: N \rightarrow V \cup \{0, 1\}$ 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, \dots, x_m ,即 $x_1 < x_2 < \dots < x_m$ 。约定对都任意变量 x 对于常量0,1都有 $x < 0$ 且 $x < 1$ 。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 $L(n') = L(n'') = x_{i+1}, L'(n') = (L'(n))_{x_i}^0, L'(n'') = (L'(n))_{x_i}^1$ 。
若 $i = m$,则令 $L(n') = L'(n') = (L'(n))_{x_m}^0, L(n'') = L'(n'') = (L'(n))_{x_m}^1$ 。
-

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 $x_1, x_3, \dots, x_{2n-1}, x_2, x_4, \dots, x_{2n}$ 。则 φ 的ROBDD节点数为 2^{n+1} 。给定变量的排序 x_1, x_2, \dots, 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_x^i 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 $f_{OBDD}(x_1, \dots, x_n)$ 表示与布尔函数 $f(x_1, \dots, x_n)$ 对应的有序二元决策图。设 $f_{OBDD}(x_1, \dots, x_n) = \langle N, n_0, E, L \rangle$ 且 $x_1 < \dots < x_n$ 。则

-
- $f_{OBDD}(0, x_2, \dots, x_n) = \langle N', n'_0, E', L' \rangle$ 其中
 N' 是由 $\pi_0(E(n_0))$ 可达的所有节点的集合； $n'_0 = \pi_0(E(n_0))$ ； $E' = E|_{N'}$ ； $L' = L|_{N'}$ 。
 - $f_{OBDD}(x_1, \dots, x_{i-1}, 0, x_{i+1}, x_n) = \langle N', n_0, E', L' \rangle$ 其中
 N' 是由 n_0 通过 $E_{x_i}^0$ 可达的所有节点的集合； $E' = E_{x_i}^0|_{N'}$ ； $L' = L|_{N'}$ 。
 - $f_{OBDD}(1, x_2, \dots, x_n) = \langle N', n'_0, E', L' \rangle$ 其中
 N' 是由 $\pi_1(E(n_0))$ 可达的所有节点的集合； $n'_0 = \pi_1(E(n_0))$ ； $E' = E|_{N'}$ ； $L' = L|_{N'}$ 。
 - $f_{OBDD}(x_1, \dots, x_{i-1}, 1, x_{i+1}, x_n) = \langle N', n_0, E', L' \rangle$ 其中
 N' 是由 n_0 通过 $E_{x_i}^1$ 可达的所有节点的集合； $E' = E_{x_i}^1|_{N'}$ ； $L' = L|_{N'}$ 。
-

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 $\exists y_1 \dots y_n. \exists x. f = \exists y_1 \dots y_n. (f|_{x=0} \vee f|_{x=1})$ 。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 $\wedge, \vee, \rightarrow, \leftrightarrow$ 。一元运算 \neg 原则上可用 \rightarrow 实现，即 $\neg p \Leftrightarrow (p \rightarrow \text{false})$ ，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, \dots, x_n 。则 $ex(\varphi) = \exists x'_1 \dots x'_n. (R(\bar{x}, \bar{x}') \wedge \varphi_{\bar{x}}^{\bar{x}'})$ 。

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。以下分别对这两种情况进行分析。

析取关系

设 $R(\bar{x}, \bar{x}') = R_1(\bar{x}, \bar{x}') \vee \dots \vee R_m(\bar{x}, \bar{x}')$ 。应用等式 $\exists x. (p(x) \vee q(x)) = \exists x. p(x) \vee \exists x. q(x)$ ，在一些情况下可以简化ROBDD的计算。我们有

$$\exists x'_1 \dots x'_n. (R(\bar{x}, \bar{x}') \wedge \varphi_{x'_1, \dots, x'_n}^{x'_1, \dots, x'_n}) = \bigvee_{i=1}^n (\exists x'_1 \dots x'_n. (R_i(\bar{x}, \bar{x}') \wedge \varphi_{\bar{x}}^{\bar{x}'}))$$

合取关系

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

若 x 不在 $p(x)$ 中出现，我们有 $\exists x. (p(x) \wedge q(x)) = p(x) \wedge \exists x. q(x)$ 。应用该等式在一些情况下可以简化ROBDD的计算。为方便起见，用 $R_0(\bar{x}, \bar{x}')$ 表示 $\varphi_{x'_1, \dots, x'_n}^{x'_1, \dots, x'_n}$ 。则

$$\exists x'_1 \dots x'_n. (R(\bar{x}, \bar{x}') \wedge \varphi_{x'_1, \dots, x'_n}^{x'_1, \dots, x'_n}) = \exists x'_1 \dots x'_n. (R_0(\bar{x}, \bar{x}') \wedge \dots \wedge R_m(\bar{x}, \bar{x}'))$$

将 x'_1, \dots, x'_n 分成 k 个不相交子集 d_1, \dots, d_k , 将 R_0, \dots, R_m 分成 k 个不相交非空子集 D_1, \dots, 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

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

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 $\{R_0, \dots, R_m\}$ 和变量集合 $\{x'_1, \dots, x'_n\}$, 我们可以依据不同的标准来计算 d_1, \dots, d_k 和 D_1, \dots, D_k 。原则上我们应该有外层的 d_i 越小而 D_i 越大越好。

§8.5.2 动态模型检测

给定一个基于谓词逻辑的迁移系统模型 M 和一个PLTL性质 φ 。由于计算效率的问题, 验证这样的问题不用先构造系统的完整的Kripke结构模型。计算 $B_M \cap B_{\neg\varphi}$ 是否为空的算法可以在 $B_{\neg\varphi}$ 已经存在的情况下, 边构造 B_M 边计算 $B_M \cap B_{\neg\varphi}$ 。一个有利的因素是在根据系统模型的高层描述构造当前状态的后继状态时, 可以判断这些新的状态是否与 $B_{\neg\varphi}$ 有共同部分, 然后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 $B_M \cap B_{\neg\varphi}$ 是否为空, 因此找到一条满足 $B_M \cap B_{\neg\varphi}$ 的运行就可退出, 不必一定等到完成整个 B_M 的构造。

踏步等价 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 $0 = i_0 < i_1 < i_2 < \dots$ 和 $0 = j_0 < j_1 < j_2 < \dots$ 使得对任意 $k \geq 0$,

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

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 $PLTL(U)$ 。该集合的公式都是踏步不变的。

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

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是根据一些判断条件, 在动态模型检测中避免类似于 s'_{i+1} 这样的状态的生成以及对 π' 的检查。

§8.6 例子

例 8.1 给定 $AP = \{p, q\}$ 和标号Kripke模型 $K = \langle S, R, I, L \rangle$ 其中 S, R, I, L 定义如下。

$$\begin{aligned} S &= \{s_0, s_1, \dots, s_{15}\}。 \\ R &= \{(s_2, s_{10}), (s_{10}, s_{10})\} \cup \{(s_i, s_i) \mid 0 \leq i \leq 15, i \notin \{2, 10\}\}。 \\ I &= \{s_2\}。 \\ L(0) &= \dots = L(3) = \{p\}; L(4) = \dots = L(7) = \{q\}; \\ L(8) &= \dots = L(11) = \{p, q\}; L(12) = \dots = L(15) = \{\}。 \end{aligned}$$

设 $V = \{v_1, v_2, v_3, v_4\}$ 。

定义 $f_0(s) = (f_0(s, 1), f_0(s, 2), f_0(s, 3), f_0(s, 4))$ 其中 $f_0(s_i, k)$ 的值为 i 写成二进制的第 k 位 (最高位为第 1 位) 的值, 定义如下。

$$\begin{aligned} f_0(s_i, 1) &= i/8; \\ f_0(s_i, 2) &= (i\%8)/4; \\ f_0(s_i, 3) &= (i\%4)/2; \\ f_0(s_i, 4) &= (i\%2); \end{aligned}$$

由以上定义知:

$(\neg v_1 \wedge \neg v_2 \wedge v_3 \wedge \neg v_4)$ 表示状态 s_2 ;

$(v_1 \wedge \neg v_2 \wedge v_3 \wedge \neg v_4)$ 表示状态 s_{10} ;

$(\neg v_2 \wedge v_3 \wedge \neg v_4)$ 表示状态集合 $\{s_2, s_{10}\}$;

$\rho_0 = (v'_1 \leftrightarrow v_1) \wedge (v'_2 \leftrightarrow v_2) \wedge (v'_3 \leftrightarrow v_3) \wedge (v'_4 \leftrightarrow v_4)$ 表示迁移集合 $\{(s_i, s_i) \mid 0 \leq i \leq 15\}$;

$\rho_1 = (\neg v_2 \wedge v_3 \wedge \neg v_4 \wedge v'_1 \wedge \neg v'_2 \wedge v'_3 \wedge \neg v_4)$ 表示迁移集合 $\{(s_2, s_{10}), (s_{10}, s_{10})\}$ 。

与 K 相应的基于 f_0 的符号模型为 $M = \langle V, \rho, \Theta, N \rangle$ 其中 ρ, Θ, N 定义如下。

$$\begin{aligned} \rho &= (\rho_1) \vee (\neg(\neg v_2 \wedge v_3 \wedge \neg v_4) \wedge \rho_0); \\ \Theta &= (\neg v_1 \wedge \neg v_2 \wedge v_3 \wedge \neg v_4); \\ N(p) &= (\neg v_2); \\ N(q) &= (\neg v_1 \wedge v_2) \vee (v_1 \wedge \neg v_2); \end{aligned}$$

例 8.2 设 $M = \langle V, \rho, \Theta, N \rangle$ 如例 8.1 中所定义。设 $\varphi = (v_1 \wedge \neg v_2 \wedge v_3 \wedge \neg v_4)$ 。

(1) $ex(\varphi)$ 和 $xe(\varphi)$ 的计算如下。

$$\begin{aligned} ex(\varphi) &= \exists v'_1 v'_2 v'_3 v'_4. (\rho \wedge \varphi') = (\neg v_2 \wedge v_3 \wedge \neg v_4) \\ xe(\varphi) &= (\exists v_1 v_2 v_3 v_4. (\rho \wedge \varphi))^\# = (v_1 \wedge \neg v_2 \wedge v_3 \wedge \neg v_4) \end{aligned}$$

(2) 判断模型是否满足 $AG(p \vee q)$ 的符号模型检测过程如下。

(a) 我们有 $[[AG(p \vee q)]] = [[\neg E(true \ U \ \neg(p \vee q))]] = \neg \mu Z. ([[\neg(p \vee q)]] \vee ex(Z))$ 。

(b) 记 $f(Z) = [[\neg(p \vee q)]] \vee ex(Z)$ 。我们有

$$\begin{aligned} Z_0 &\equiv false \\ f^1(Z_0) &\equiv [[\neg(p \vee q)]] = (v_1 \wedge v_2) \\ f^2(Z_0) &\equiv (v_1 \wedge v_2) \end{aligned}$$

因而 $[[AG(p \vee q)]] = \neg(v_1 \wedge v_2)$ 。

(c) 由 $\Theta \rightarrow \neg(v_1 \wedge v_2)$ 知模型满足 $AG(p \vee q)$ 。

§8.7 练习

1. 给定 $AP = \{p, q\}$ 和标号 Kripke 模型 $K = \langle S, R, I, L \rangle$ 其中 S, R, I, L 定义如下。

$$\begin{aligned} S &= \{s_0, s_1, s_2, s_3\}。 \\ R &= \{(s_0, s_1), (s_0, s_2), (s_1, s_3), (s_2, s_0), (s_2, s_3), (s_3, s_1)\}。 \\ I &= \{s_0, s_2\}。 \\ L(s_0) &= \{\}; \quad L(s_1) = \{q\}; \quad L(s_2) = \{p\}; \quad L(s_3) = \{p, q\}。 \end{aligned}$$

将 K 转换为符号模型。

2. 给定 $AP = \{p, q\}$ 上的符号模型 $M = \langle V, \rho, \Theta, N \rangle$ 其中 V, ρ, Θ, N 定义如下。

$$\begin{aligned} V &= \{v_1, v_2\}; \\ \rho &= (\neg v_1 \wedge \neg v_2 \wedge (v'_1 \leftrightarrow \neg v'_2)) \vee (\neg v_1 \wedge v_2 \wedge (v'_1 \wedge v'_2)) \vee \\ &\quad (v_1 \wedge \neg v_2 \wedge (v'_1 \leftrightarrow v'_2)) \vee (v_1 \wedge v_2 \wedge (\neg v'_1 \wedge v'_2)); \\ \Theta &= (\neg v_2); \\ N(p) &= (v_1); \\ N(q) &= (v_2); \end{aligned}$$

- (1) 计算 v_2 的前驱状态集公式 $ex(v_2)$ 和后继状态集公式 $xe(v_2)$ 。
- (2) 用符号模型检测方法计算 $[[EG(p \vee q)]]$ 并说明模型是否满足 $EG(p \vee q)$ 。
- (3) 阐述用限界正确性检查方法判断模型是否满足 $EG(p \vee q)$ 的计算过程。

§9 实例分析

本章介绍实例。主要实例为互斥算法、窗口协议与整数平方根算法。

§9.1 互斥算法

本节考虑两个进程的互斥算法。

互斥算法的卫式迁移模型： 给定 $B = (\{s_0, s_1, s_2, s_3, t_0, t_1, t_2, t_3, 0, 1\}, \{=\})$ 。给定 $I = (D, I_0)$ 其中 $D = \{0, 1, 2, 3\}$ 且 I_0 定义如下：

$I_0(=) =$	$=$
$I_0(s_0) = I_0(t_0) = I(0) =$	0
$I_0(s_1) = I_0(t_1) = I(1) =$	1
$I_0(s_2) = I_0(t_2) =$	2
$I_0(s_3) = I_0(t_3) =$	3

给定 $V = \{a, b, x, y, t\}$ 。

(B, V) 上的互斥算法的卫式迁移模型定义为 $M = \langle T, \Theta \rangle$ 其中 Θ 为 $a = s_0 \wedge b = t_0 \wedge x = 0 \wedge y = 0$ 且 T 为以下迁移的集合：

$a = s_0$	\longrightarrow	$(y, t, a) := (1, 1, s_1)$
$a = s_1 \wedge (x = 0 \vee t = 0)$	\longrightarrow	$(a) := (s_2)$
$a = s_2$	\longrightarrow	$(y, a) := (0, s_3)$
$a = s_3$	\longrightarrow	$(y, t, a) := (1, 1, s_1)$
$b = t_0$	\longrightarrow	$(x, t, b) := (1, 0, t_1)$
$b = t_1 \wedge (y = 0 \vee t = 1)$	\longrightarrow	$(b) := (t_2)$
$b = t_2$	\longrightarrow	$(x, b) := (0, t_3)$
$b = t_3$	\longrightarrow	$(x, t, b) := (1, 0, t_1)$

该模型状态迁移部分的示意图见图2。 $a = s_i$ 表示进程A处于 s_i 状态。 s_0, s_1, s_2, s_3 分别理解为进程A的初始状态，申请状态，使用资源状态，普通状态。类似地， $b = t_i$ 表示进程B处于 t_i 状态。

用五元组 (a, b, x, y, t) 表示系统状态。系统总状态数为 $4 * 4 * 2 * 2 * 2 = 128$ 个。以下是系统的可达状态，其中前两个为初始状态。

- $(s_0, t_0, 0, 0, 0)$ $(s_0, t_0, 0, 0, 1)$
- $(s_1, t_0, 0, 1, 1)$ $(s_0, t_1, 1, 0, 0)$
- $(s_2, t_0, 0, 1, 1)$ $(s_1, t_1, 1, 1, 0)$
- $(s_1, t_1, 1, 1, 1)$ $(s_0, t_2, 1, 0, 0)$
- $(s_3, t_0, 0, 0, 1)$ $(s_2, t_1, 1, 1, 0)$
- $(s_1, t_2, 1, 1, 1)$ $(s_0, t_3, 0, 0, 0)$
- $(s_3, t_1, 1, 0, 0)$ $(s_1, t_3, 0, 1, 1)$
- $(s_3, t_2, 1, 0, 0)$ $(s_2, t_3, 0, 1, 1)$
- $(s_3, t_3, 0, 0, 0)$ $(s_3, t_3, 0, 0, 1)$

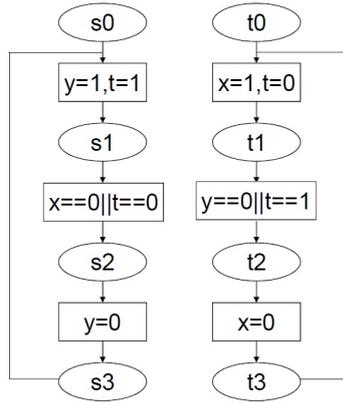


图 1 互斥算法示意图

互斥算法的性质： 我们考虑以下三个性质。

$$\begin{aligned} \varphi_1 &= \Box(\neg(a = s_2 \wedge b = t_2)). \\ \varphi_2 &= \Box(a = s_1 \wedge \neg(b = t_1 \vee b = t_2) \rightarrow (a = s_2 R b \neq t_2)). \\ \varphi_3 &= \Box(a = s_1 \rightarrow \Diamond a = s_2). \end{aligned}$$

这三个PLTL性质亦可用CTL表示如下。

$$\begin{aligned} \varphi'_1 &= AG(\neg(a = s_2 \wedge b = t_2)). \\ \varphi'_2 &= AG(a = s_1 \wedge \neg(b = t_1 \vee b = t_2) \rightarrow A(a = s_2 R b \neq t_2)). \\ \varphi'_3 &= AG(a = s_1 \rightarrow AF a = s_2). \end{aligned}$$

§9.1.1 演绎推理

(a) $M \models_I \varphi_1$ 的证明如下。

拟应用§7.1节的证明结论形式为 $\varphi \Rightarrow \Box\psi$ 的如下推理规则。

$$\frac{\begin{array}{l} \varphi \Rightarrow \phi \\ \phi \rightarrow [T]\phi \\ \phi \rightarrow \psi \end{array}}{\varphi \Rightarrow \Box\psi}$$

首先证明 $M \models_I \Box(\Theta \rightarrow \varphi_1)$ 如下。

定义 $\varphi = \Theta$ 且 $\psi = (\neg(a = s_2 \wedge b = t_2))$ 。

定义 ϕ 为以下公式的析取。

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

则有 $\varphi \Rightarrow \phi$ 且 $\phi \rightarrow [T]\phi$ 且 $\phi \Rightarrow \psi$ 。

因而根据证明规则我们有 $\varphi \Rightarrow \Box\psi$ ，即 $M \models_I \Box(\Theta \rightarrow \varphi_1)$ 。

因而 $M \models_I \Theta \rightarrow \varphi_1$ 。进而我们有 $M \models_I \varphi_1$ 。

(b) $M \models_I \Box(a = s_1 \wedge b \neq t_1 \wedge b \neq t_2 \rightarrow (a = s_2 R b \neq t_2))$ 的证明如下。

设 $\varphi = (a = s_1 \wedge b \neq t_1 \wedge b \neq t_2)$, $\varphi_0 = (a = s_2)$, $\psi = (b \neq t_2)$ 。

拟应用以下推理规则证明 $\varphi \Rightarrow \varphi_0 R \psi$ 。

$$\begin{array}{l} \varphi \Rightarrow \phi \\ (\phi \wedge \neg\varphi_0) \rightarrow [T]\phi \\ \phi \rightarrow \psi \\ \hline \varphi \Rightarrow \varphi_0 R \psi \end{array}$$

我们需定义 ϕ 使得推理规则中的三个前提条件成立。

定义 ϕ 为以下公式。

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

推理规则中的第一个和第三个条件显然成立。类似于例7.2中的计算可得 $[T](b \neq t_2)$ 的计算结果并由此易知第二个条件亦成立，因而所需证明的结论成立。

(c) $M \models_I \Box(a = s_1 \rightarrow \Diamond(a = s_2))$ 的证明如下。

(c1) 由于证明中需要用到 $M \models_I \Box(a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ ，我们先证明这个性质如下。

若有 $\Theta \Rightarrow \Box(a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ ，则有 $M \models_I \Theta \rightarrow \Box(a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ ，然后由 Θ 规则可得 $M \models_I \Box(a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ 成立。

以下证明 $\Theta \Rightarrow \Box(a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ 。

设 $\varphi = \Theta$, $\psi = (a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ 。

拟应用 (a) 中用到的推理规则证明 $\varphi \Rightarrow \Box\psi$ 。

我们需定义 ϕ 使得推理规则中的三个前提条件成立。

定义 ϕ 为以下公式。

$$(a = s_1 \rightarrow y = 1)$$

推理规则中的第一个和第三个条件显然成立。通过计算 $[T]\phi$ 知第二个条件亦成立，因而 $\Theta \Rightarrow \Box(a = s_1 \rightarrow (a = s_1 \wedge y = 1))$ 成立。

(c2) 性质 $M \models_I \Box(a = s_1 \rightarrow \Diamond(a = s_2))$ 的证明如下。

设 $\varphi = (a = s_1)$, $\psi = (a = s_2)$ 。

拟应用以下推理规则证明 $\varphi \Rightarrow \Diamond\psi$ 。

$$\begin{array}{l} \varphi \Rightarrow (\psi \vee \phi) \\ \phi \rightarrow (w_x^e \wedge (E(T) \vee \psi)) \\ (\phi \wedge e = v) \rightarrow [T](\psi \vee (\phi \wedge e \sqsubset v)) \\ \hline \varphi \Rightarrow \Diamond\psi \end{array}$$

对 $B = (F, P)$ 进行扩充, 增加函数符号 h , 谓词符号 \sqsubseteq , 相应扩充 I_0 使得 \sqsubseteq 解释为通常自然数上的大小关系且 h 有解释如下。

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

我们需定义 e, w, ϕ 使得 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, I_0(\sqsubseteq))$ 为良基集合且推理规则中的三个前提条件成立。

定义

$$\begin{aligned} e &= h(b, t) \\ w &= (0 \sqsubseteq x \sqsubseteq 3) \\ \phi &= (a = s_1 \wedge y = 1) \end{aligned}$$

易知 $W = \{0, 1, 2, 3\}$ 为良基集合。

由(c1)知推理规则中的第一个条件成立。易知第二个条件成立。通过计算 $[T](\psi \vee (\phi \wedge e \sqsubseteq v))$ 知第三个条件亦成立, 因而所需证明的结论成立。

§9.1.2 基于模型检测算法的验证

假定我们依然验证 φ_1 和 φ_2 这两个性质。那么我们关心的命题的集合为 $\{a = s_1, a = s_2, b = t_1, b = t_2\}$ 。分别用 p_1, p_2, q_1, q_2 表示这些命题。定义 $AP = \{p_1, p_2, q_1, q_2\}$ 。

定义 $a(i) = (i/32)$, $b(i) = (i/8)\%4$, $x(i) = (i/4)\%2$, $y(i) = (i/2)\%2$, $t(i) = i\%2$ 。

定义 M 的标号Kripke模型 $(M)^{km} = \langle S, R, I, L \rangle$ 的各分量如下。

$$S = \{w_i \mid 0 \leq i \leq 127\}。$$

$R = R_1 \cup \dots \cup R_8$ 其中 R_1, \dots, R_8 定义如下。

$$R_1 = \{(w_i, w_j) \mid a(i) = 0, y(j) = 1, t(j) = 1, a(j) = 1\}$$

$$R_2 = \{(w_i, w_j) \mid a(i) = 1, x(i) = 0 \vee t(i) = 0, a(j) = 2\}$$

$$R_3 = \{(w_i, w_j) \mid a(i) = 2, y(j) = 0, a(j) = 3\}$$

$$R_4 = \{(w_i, w_j) \mid a(i) = 3, y(j) = 1, t(j) = 1, a(j) = 1\}$$

$$R_5 = \{(w_i, w_j) \mid b(i) = 0, x(j) = 1, t(j) = 0, b(j) = 1\}$$

$$R_6 = \{(w_i, w_j) \mid b(i) = 1, y(i) = 0 \vee t(i) = 1, b(j) = 2\}$$

$$R_7 = \{(w_i, w_j) \mid b(i) = 2, x(j) = 0, b(j) = 3\}$$

$$R_8 = \{(w_i, w_j) \mid b(i) = 3, x(j) = 1, t(j) = 0, b(j) = 1\}$$

$$I = \{w_0, w_1\}。$$

L 定义如下。

对任意 $i \in \{0, \dots, 127\}$, $k \in \{1, 2\}$, $p_k \in L(w_i)$ 当且仅当 $a(i) = k$ 。

对任意 $i \in \{0, \dots, 127\}$, $k \in \{1, 2\}$, $q_k \in L(w_i)$ 当且仅当 $b(i) = k$ 。

我们有以下结论。

$M \models_I \varphi_1$ 当且仅当 $(M)^{km} \models \Box(\neg(p_2 \wedge q_2))$
$M \models_I \varphi_2$ 当且仅当 $(M)^{km} \models \Box(p_1 \wedge \neg(q_1 \vee q_2) \rightarrow (p_2 R \neg q_2))$
$M \models_I \varphi_3$ 当且仅当 $(M)^{km} \models \Box(p_1 \rightarrow \Diamond p_2)$
$M \models_I \varphi'_1$ 当且仅当 $(M)^{km} \models AG(\neg(p_2 \wedge q_2))$
$M \models_I \varphi'_2$ 当且仅当 $(M)^{km} \models AG(p_1 \wedge \neg(q_1 \vee q_2) \rightarrow A(p_2 R \neg q_2))$
$M \models_I \varphi'_3$ 当且仅当 $(M)^{km} \models AG(p_1 \rightarrow AFP_2)$

模型检测算法过程举例： 由于计算的繁琐，仅举一例，即用不动点算法证明 $(M)^{km} \models AG(\neg(p_2 \wedge q_2))$ ，即证明 $(M)^{km} \models \neg EF(p_2 \wedge q_2)$ 如下。

- 我们有 $EF(p_2 \wedge q_2) = \mu Z.((p_2 \wedge q_2) \vee EXZ)$ 。
- 为方便书写， $a(i), b(i), x(i), y(i), t(i)$ 分别写成 a, b, x, y, t 。
定义 f_i 为以下表格中行数标号为数字 i 的状态集合。

标号	状态集合	说明
1	$\{w_i \mid a = 2 \wedge b = 2\}$	
11	$\{w_i \mid a = 1 \wedge (x = 0 \vee t = 0) \wedge b = 2\}$	
12	$\{w_i \mid a = 2 \wedge b = 1 \wedge (y = 0 \vee t = 1)\}$	
111	$\{w_i \mid (a = 0 \vee a = 3) \wedge x = 0 \wedge b = 2\}$	
112	$\{w_i \mid a = 1 \wedge (x = 0 \vee t = 0) \wedge b = 1 \wedge (y = 0 \vee t = 1)\}$	
121	$\{w_i \mid a = 1 \wedge (x = 0 \vee t = 0) \wedge b = 1 \wedge (y = 0 \vee t = 1)\}$	等同于112
122	$\{w_i \mid a = 2 \wedge (b = 0 \vee b = 3) \wedge y = 0\}$	
1111	$\{w_i \mid a = 2 \wedge x = 0 \wedge b = 2\}$	包含于1
1112	$\{w_i \mid (a = 0 \vee a = 3) \wedge x = 0 \wedge b = 1 \wedge (y = 0 \vee t = 1)\}$	
1121	$\{w_i \mid (a = 0 \vee a = 3) \wedge x = 0 \wedge b = 1 \wedge (y = 0 \vee t = 1)\}$	等同于1112
1122	$\{w_i \mid a = 1 \wedge (x = 0 \vee t = 0) \wedge (b = 0 \vee b = 3) \wedge y = 0\}$	等同于1221
1221	$\{w_i \mid a = 1 \wedge (x = 0 \vee t = 0) \wedge (b = 0 \vee b = 3) \wedge y = 0\}$	
1222	$\{w_i \mid a = 2 \wedge b = 2 \wedge y = 0\}$	包含于1
11121	$\{w_i \mid a = 2 \wedge x = 0 \wedge b = 1 \wedge (y = 0 \vee t = 1)\}$	包含于12
11122	$\{\}$	
12211	$\{\}$	
12212	$\{w_i \mid a = 1 \wedge (x = 0 \vee t = 0) \wedge b = 2 \wedge y = 0\}$	包含于11

- 我们有 $(p_2 \wedge q_2) = \{w_i \mid a(i) = 2 \wedge b(i) = 2\} = f_1$ 。
- 设 $\tau(Z) = (p_2 \wedge q_2) \vee EXZ$ 。则 $\mu Z. \tau(Z)$ 的计算如下。

$$\tau^0(false) = false;$$

$$\tau^1(false) = f_1;$$

$$\tau^2(false) = \tau^1(false) \cup f_{11} \cup f_{12};$$

$$\tau^3(false) = \tau^2(false) \cup f_{111} \cup f_{112} \cup f_{122};$$

$$\tau^4(false) = \tau^3(false) \cup f_{1112} \cup f_{1221};$$

$$\tau^5(false) = \tau^4(false);$$

$$\text{因此 } EF(p_2 \wedge q_2) = \mu Z.((p_2 \wedge q_2) \vee EXZ) = \mu Z.\tau(Z) = \tau^4(false)。$$

$$\text{因此 } AG(\neg(p_2 \wedge q_2)) = \neg EF(p_2 \wedge q_2) = S \setminus \tau^4(false)。$$

- 由 $\tau^4(false)$ 不含有 $(M)^{km}$ 的初始状态知 $I \subseteq AG(\neg(p_2 \wedge q_2))$ ，即 $(M)^{km} \models AG(\neg(p_2 \wedge q_2))$ 。
因此我们由 $M \models_I \varphi'_1$ 。

§9.1.3 基于模型检测工具的自动验证

鉴于手工验证的繁琐，我们可使用模型检测工具进行验证。

应用VERDS建模语言VML建立VERDS验证模型如下。

```

/*****
VVM      mutex01.vvm
VAR      a: {s0,s1,s2,s3}; b: {t0,t1,t2,t3}; x: 0..1; y: 0..1; t: 0..1;
INIT     a=s0; b=s0; x=0; y=0;
TRANS    a=s0:                (y,t,a):=(1,1,s1);
          a=s1&(x=0|t=0):     (a):=(s2);
          a=s2:                (y,a):=(0,s3);
          a=s3:                (y,t,a):=(1,1,s1);
          b=t0:                (x,t,b):=(1,0,t1);
          b=t1&(y=0|t=1):     (b):=(t2);
          b=t2:                (x,b):=(0,t3);
          b=t3:                (x,t,b):=(1,0,t1);
SPEC     AG(!(a=s2&b=s2));
          AG(!(a=s1&!(b=t1|b=t2))|A(a=s2 R b!=t2));
          AG!(a=s1|AF(a=s2));
*****/

```

用VERDS验证 $\varphi'_1, \varphi'_2, \varphi'_3$ 的结果说明这些性质成立。

§9.1.4 公平约束模型的模型检测

假定我们的模型增加两条以下迁移，记为 T_1 。

$$\begin{array}{l} a = s_1 \wedge \neg(x = 0 \vee t = 0) \longrightarrow (a) := (s_1) \\ b = t_1 \wedge \neg(y = 0 \vee t = 1) \longrightarrow (b) := (t_1) \end{array}$$

这两条迁移表示当进程A在 s_1 状态时对条件 $(x = 0 \vee t = 0)$ 进行检查且条件不满足时回到 s_1 状态（模型中这种迁移理解为原地踏步地走了一步），同样，进程B在 t_1 状态时可以回到 t_1 这个状态。记修改后的模型为 $M' = \langle T \cup T_1, \Theta \rangle$ 。

那么修改后的模型满足 φ_1 和 φ_2 ，不满足 φ_3 。

应用VERDS建模语言VML建立VERDS验证模型并验证 $\varphi'_1, \varphi'_2, \varphi'_3$ ，验证结果说明前两个性质成立和后一个性质不成立。

公平约束： 以 M' 为出发点，我们要求若进程A一直处在 s_0 状态那么A必须能完成一次状态迁移；若A一直处在 s_1 状态且条件 $(x = 0 \vee t = 0)$ 一直成立那么A必须对条件进行检查并完成一次状态迁移；若A一直处在 s_2 状态那么A必须完成一次状态迁移；若A一直处在 s_3 状态那么A必须完成一次状态迁移。相应的，B进程也是类似。这个可以表示为公平约束集合 $\Phi = \{\neg(a = s_0), \neg(a = s_1 \wedge (x = 0 \vee t = 0)), \neg(a = s_2), \neg(a = s_3), \neg(b = t_0), \neg(b = t_1 \wedge (y = 0 \vee t = 1)), \neg(b = t_2), \neg(b = t_3)\}$ 。那么这个公平卫式迁移模型为 $M'' = \langle T \cup T_1, \Theta, \Phi \rangle$ 。

应用VERDS建模语言VML的模块化建模功能建立VERDS验证模型如下。

```

/*****/
VVM      mutex02.vvm
VAR      x: 0..1; y: 0..1; t: 0..1;
INIT     x=0; y=0;
PROC     p0: m0(x,y,0); p1: m0(y,x,1);
SPEC     AG(!(p0.a=s2&p1.a=s2));
          AG(!(p0.a=s1&!(p1.a=s1|p1.a=s2)) | A(p0.a=s2 R p1.a!=s2));
          AG(!(p0.a=s1) | AF(p0.a=s2));
MODULE  m0(x0,y0,k)
VAR      a: {s0,s1,s2,s3};
INIT     a=s0;
TRANS    a=s0:                (y0,t,a):=(1,1-k,s1);
          a=s1&(x0=0|t=k):    (a):=(s2);
          a=s1&!(x0=0|t=k):  (a):=(s1);
          a=s2:                (y0,a):=(0,s3);
          a=s3:                (y0,t,a):=(1,1-k,s1);
FAIRNESS ! (a=s0);
          ! (a=s1&(x0=0|t=k));
          ! (a=s2);
          ! (a=s3);
/*****/

```

应用VERDS验证 $\varphi'_1, \varphi'_2, \varphi'_3$ ，验证结果说明这些性质在新模型下成立，即 $M'' \models_I \varphi'_1$ ， $M'' \models_I \varphi'_2$ 和 $M'' \models_I \varphi'_3$ 。对于这些性质而言，模块定义中的第二条和第三条约束保证了 φ'_3 的正确性。

§9.2 窗口协议

本节考虑一个稍微复杂一点的协议（窗口协议）的建模与模型检测。

设 win 为常量（参数），表示窗口长度。 $nbox = 2win$ 为常量，表示接收方的缓存个数。为方便起见使用长度为 $nbox$ 的数组表示缓存。用 $p0$ 表示发送方， $p1$ 表示接收方。通道相关变量的说明如下。

$p1chanlen$	取值范围 $\{0, 1, \dots, win\}$, 表示通道 $p1chan$ ($p1$ 可读取) 内的有效信息个数。
$p1chanidx[]$	长度为 win 的数组, 取值范围 $\{0, 1, \dots, nbox - 1\}$, 表示通道 $p1chan$ 中信息的序号。
$p1chancont[]$	长度为 win 的数组, 取值范围 $\{0, 1\}$, 表示通道 $p1chan$ 中信息的内容, 在模型中仅表示内容是否在发送过程中出现错误, 0表示错误。
$p1channext$	取值范围 $\{0, 1, win - 1\}$, 指向通道 $p1chan$ 的第一个空位。
$p0chanlen$	取值范围 $\{0, 1\}$, 表示通道 $p0chan$ ($p0$ 可读取) 内的有效信息个数, 0或1。
$p0chanidx$	取值范围 $\{0, 1, \dots, nbox - 1\}$, 表示通道 $p0chan$ 中信息的内容 (仅为一个序号), 序号的值表示 $p1$ 已经得到该序号之前的所有内容。

发送方与接收方变量的说明如下。

$p0s$	取值范围 $\{s_0, s_1, s_2\}$, 表示 $p0$ 所处的控制状态。
x	取值范围 $\{0, 1, \dots, win - 1\}$, 表示 $p1$ 等待的下一个信息的序号。
k	取值范围 $\{0, 1, \dots, win - 1\}$, $(x + k) \% win$ 为 $p0$ 要发送的序号。
$p1s$	取值范围 $\{r_0, r_1, \dots, r_4\}$, 表示 $p1$ 所处的控制状态。
$box[]$	长度为 $nbox$ 的数组, 取值范围 $\{0, 1\}$, $box[i] = 1$ 表示序号为 i 的消息已收到, 等待进一步处理。
y	取值范围 $\{0, 1, \dots, nbox - 1\}$, 表示 $p1$ 已经得到该序号之前的内容。
n	取值范围 $\{0, 1, \dots, nbox - 1\}$, 用以临时存储 $p1$ 得到的信息的序号。
c	取值范围 $\{0, 1\}$, 用以临时存储 $p1$ 得到的信息的内容。

给定 $B = (\{s_0, s_1, s_2, r_0, r_1, r_2, r_3, r_4, 0, 1, \dots, win, +, -, *, /, \%, \{>, \geq, =, <, \leq\})$ 。给定 $I = (D, I_0)$ 其中 $D = \{0, 1, 2, \dots\}$ 且 I_0 对部分符号的定义如下:

$I_0(=)$	=
$I_0(r_0) = I_0(s_0) = I(0)$	0
$I_0(r_1) = I_0(s_1) = I(1)$	1
$I_0(r_2) = I_0(s_2) = I(2)$	2
$I_0(r_3)$	3
$I_0(r_4)$	4

其余符号类似于这些符号在自然数域上的定义。给定 V 如上所述。

为阅读方便, 用 $p1head$ 和 $p1last$ 分别代表通道 $p1chan$ 的第一个有内容的位置和最后一个有内容的位置 (若通道确实有有效内容), 定义如下。

$$p1head = (p1channext + (win - p1chanlen)) \% win$$

$$p1last = (p1channext + (win - 1)) \% win$$

用 $p0put(p0s, i, c, a)$ 表示 $p0$ 往 $p1$ 的通道放入 x, c , 并转入 $p0s = a$ 的状态, 定义如下。

$$(p0s, p1chanidx[p1channext], p1chancont[p1channext], p1channext, p1chanlen) := (a, (i+k)\%nbox, c, p1channext + 1, p1chanlen + 1)$$

用 $p0get(p0s, x, k, a)$ 表示 $p0$ 从自己的通道读出数据 i 并重置 k 为0, 并转入 $p0s = a$ 的状态, 定义如下。

$$(p0s, x, k, p1chanlen) := (a, p1chanidx, 0, 0)$$

用 $p1put(p1s, y, a)$ 表示 $p1$ 往 $p0$ 的通道放入 y , 并转入 $p1s = a$ 的状态, 定义如下。

$$(p1s, p0chanidx, p1chanlen) := (a, i, 1)$$

用 $p1get(p1s, n, c, a)$ 表示 $p1$ 从自己的通道读出数据 n, c , 并转入 $p1s = a$ 的状态, 定义如下。

$$(p1s, n, c, p1chanlen, p1chanidx[p1head], p1chancont[p1head]) := (a, p1chanidx[p1head], p1chancont[p1head], p1chanlen - 1, 0, 0)$$

用 $inrange(n)$ 表示 $p1$ 得到的数据 n 为在应得范围的有效数据, 定义如下。

$$(((n \geq y) \wedge (n - y < win)) \vee ((y > n) \wedge (y - n > win)))$$

§9.2.1 可靠通道的窗口协议的卫式迁移模型:

(B, V) 上的窗口协议的卫式迁移模型定义为 $M = \langle T, \Theta \rangle$ 其中 Θ 为 $p0s = s_0$ 和 $p1s = r_0$ 且其余变量初始值为0, T 为以下迁移的集合:

$p0s = s0 \wedge p1chanlen < win$	\longrightarrow	$p0put(p0s, (x+k)\%nbox, 1, s1);$
$p0s = s1 \wedge p0chanlen = 1$	\longrightarrow	$p0get(p0s, x, k, s0);$
$p0s = s1 \wedge p0chanlen = 0$	\longrightarrow	$(p0s) := (s2);$
$p0s = s2$	\longrightarrow	$(p0s, k) := (s0, (k+1)\%win);$
$p1s = r0 \wedge p1chanlen > 0$	\longrightarrow	$p1get(p1s, n, c, r1);$
$p1s = r1 \wedge (c = 1 \wedge inrange(n))$	\longrightarrow	$(p1s, box[n]) := (r2, 1);$
$p1s = r1 \wedge \neg(c = 1 \wedge inrange(n))$	\longrightarrow	$(p1s) := (r0);$
$p1s = r2 \wedge (box[y] = 1)$	\longrightarrow	$(p1s, box[y], y) := (r3, 0, (y+1)\%nbox);$
$p1s = r2 \wedge \neg(box[y] = 1)$	\longrightarrow	$(p1s) := (r0);$
$p1s = r3 \wedge (box[y] = 1)$	\longrightarrow	$(p1s, box[y], y) := (r3, 0, (y+1)\%nbox);$
$p1s = r3 \wedge \neg(box[y] = 1)$	\longrightarrow	$(p1s) := (r4);$
$p1s = r4 \wedge p0chanlen = 0$	\longrightarrow	$p1put(p1s, y, r0);$

该模型状态迁移部分的示意图见图2。 $p0s = s_i$ 表示进程 $p0$ 处于 s_i 状态。类似地, $p1s = r_i$ 表示进程 $p1$ 处于 r_i 状态。

窗口协议的性质: 考虑 win 为2的时以下三个性质。

$$\begin{aligned} \varphi_1 &= \Box(\neg(x = 3 \wedge y = 2)). \\ \varphi_2 &= \Box(x = 3 \rightarrow ((y = 1 \vee y = 2) R box[1] = 0 \wedge box[2] = 0)). \\ \varphi_3 &= \Box(x = 3 \rightarrow \Diamond(x = 0 \vee x = 1)). \end{aligned}$$

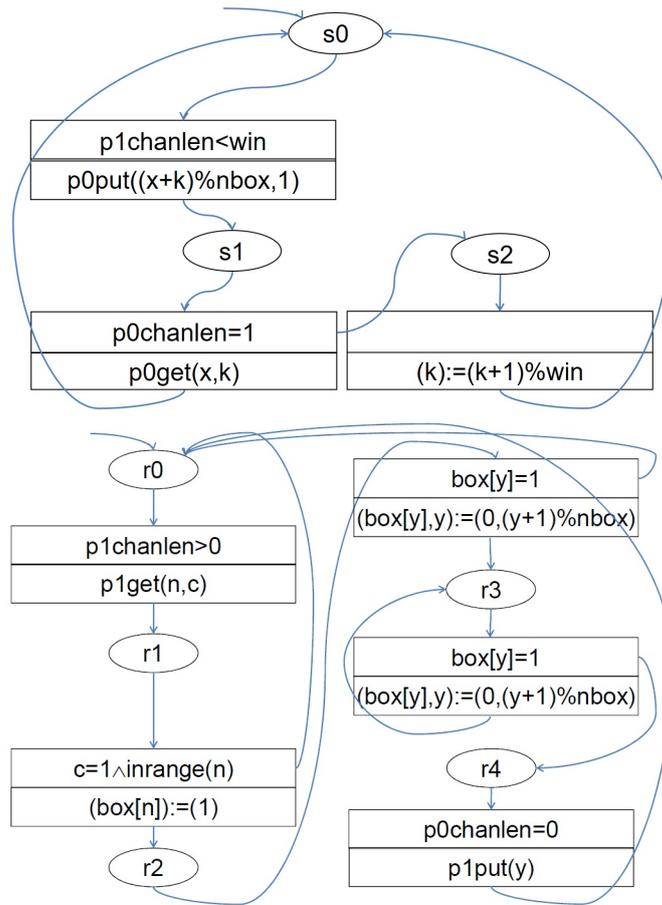


图2 窗口协议示意图

这三个PLTL性质亦可用CTL表示如下。

$$\begin{aligned} \varphi'_1 &= AG(\neg(x = 3 \wedge y = 2)). \\ \varphi'_2 &= AG(x = 3 \rightarrow A((y = 1 \vee y = 2) R box[1] = 0 \wedge box[2] = 0)). \\ \varphi'_3 &= AG(x = 3 \rightarrow AF(x = 0 \vee x = 1)). \end{aligned}$$

前面的建模假定了传输过程信息内容不会出错。那么这三个性质在模型下都成立，即我们有以下结论。

$$(T, \Theta) \models \varphi_1 \wedge \varphi_2 \wedge \varphi_3$$

基于模型检测工具的自动验证 应用VERDS建模语言VML建立窗口协议的2个进程的VERDS验证模型如下。

```

/*****
VVM
DEFINE

```

```

win=2
nbox=4
p1head=(p1channext+(win-p1chanlen))%win
p1last=(p1channext+(win-1))%win
inrange=((n>=y)&(n-y<win))|((y>n)&(y-n>win))
VAR
p1chanlen:0..2;                //win=2;
p1chanidx[0..1]:0..3;          //win-1=1; nbox-1=3;
p1chancont[0..1]:0..1;        //win-1=1; constant=1;
p1channext:0..1;              //win-1=1;
p0chanlen:0..1;               //constant=1;
p0chanidx:0..3;               //nbox-1=3;
INIT
p1chanlen=0;
for (i:[0..1]){p1chanidx[i]=0}; //win-1=1;
for (i:[0..1]){p1chancont[i]=0}; //win-1=1;
p1channext=0; p0chanlen=0; p0chanidx=0;
PROC
p0:p0m(); p1:p1m();
SPEC
AG(!(p0.x=3&p1.y=2));
AG(!(p0.x=3)|A((p1.y=0|p1.y=1)R(p1.box[1]=0&p1.box[2]=0)));
AG(!(p0.x=3)|AF(p0.x=0|p0.x=1));
MODULE p0m()
VAR
pc:{s0,s1,s2};
x:0..3;                //nbox-1=3;
k:0..1;                //win-1=1;
INIT
pc=s0; x=0; k=0;
TRANS
pc=s0&p1chanlen<win:
(pc,p1chanidx[p1channext],p1chancont[p1channext],p1channext,p1chanlen):=
(s1,(x+k)%nbox,1,(p1channext+1)%win,p1chanlen+1);
pc=s1&p0chanlen=1:      (pc,p0chanlen,x,k):=(s0,0,p0chanidx,0);
pc=s1&p0chanlen=0:      (pc):=(s2);
pc=s2:                  (pc,k):=(s0,(k+1)%win);
MODULE p1m()
VAR
pc:{r0,r1,r2,r3,r4};

```

```

y:0..3; //nbox-1=3;
n:0..3; //nbox-1=3;
c:0..1; //constant=1;
box[0..3]:0..1; //nbox-1=3; constant=1;
INIT
pc=r0; y=0; n=0; c=0;
for (i:[0..3]){box[i]=0}; //nbox-1=3;
TRANS
pc=r0&p1chanlen>0:
(pc,n,c,p1chanlen,p1chanidx[p1head],p1chancont[p1head]):=
(r1,p1chanidx[p1head],p1chancont[p1head],p1chanlen-1,0,0);
pc=r1&(c=1&inrange): (pc,box[n]):=(r2,1);
pc=r1&!(c=1&inrange): (pc):=(r0);
pc=r2&(box[y]=1): (pc,box[y],y):=(r3,0,(y+1)%nbox);
pc=r2&!(box[y]=1): (pc):=(r0);
pc=r3&(box[y]=1): (pc,box[y],y):=(r3,0,(y+1)%nbox);
pc=r3&!(box[y]=1): (pc):=(r4);
pc=r4&p0chanlen=0: (pc,p0chanlen,p0chanidx):=(r0,1,y);
/*****

```

应用VERDS验证 $\varphi'_1, \varphi'_2, \varphi'_3$ 的结果说明这些性质成立。

§9.2.2 不可靠通道的窗口协议的卫式迁移模型:

倘若传输过程信息内容可能出错, 那么我们在 p_0 的建模中加入以下迁移, 其中0表示内容出错, p_1 收到时忽略这个内容。

$$p_0s = s_0 \wedge p_1chanlen < win \longrightarrow p_0put(p_0s, (x+k)\%nbox, 0, s_1);$$

按照VERDS的写法, 就是在 p_0m 模块定义的迁移部分增加以下一段。

```

pc=s0&p1chanlen<win:
(pc,p1chanidx[p1channext],p1chancont[p1channext],p1channext,p1chanlen):=
(s1,(x+k)%nbox,0,(p1channext+1)%win,p1chanlen+1);

```

在这个模型下, 前两个性质成立, 而第三个性质不成立。用 t 表示上面这个迁移, 我们有以下结论。

$$(T \cup \{t\}, \Theta) \not\models \varphi_3$$

第三个性质不成立的原因是模型允许某个内容在传输过程中总是出错。那么我们可以用以下公平性约束说明如果 p_0 不停地传输与序号 x 相关的内容, 那么总有不出错的时候。

$$\Phi = \{\neg(\neg(p_1chanlen > 0 \wedge p_1chanidx[p_1last] = x) \vee p_1chancont[p_1last] = 0)\}$$

按照VERDS的写法, 就是在 p_0m 模块定义的最后增加以下一段。

FAIRNESS $!(!(p1chanlen > 0 \& p1chanidx[p1head] = x) | p1chancont[p1head] = 0);$

在这个公平性约束的条件下，模型满足第三个性质，即

$$(T \cup \{t\}, \Theta, \Phi) \models \varphi_3$$

§9.2.3 窗口协议的客户端模型

由于传输较多数据的通道有可能造成数据失真被丢弃，窗口协议的用途是将不可靠通道变成可靠通道。从建模来讲，我们可以增加如下两个发送数据的客户和接收数据的客户的模块（假定 $win = 2$ ）。

```

MODULE q0m()
VAR
wbox[0..1]:0..3;           //win-1=1; nbox-1=3;
cnt:0..2;                 //win=2;
INIT
for (i:[0..1]){wbox[i]=i}; //win-1=1;
cnt=win;
TRANS
cnt<2:(wbox[0],wbox[1],cnt):=(p0.x,(p0.x+1)%nbox,2); //win=2;
MODULE q1m()
VAR
idx:0..3;                 //nbox-1=3;
c:0..1;
INIT
idx=0; c=0;
TRANS
FALSE: TRUE;

```

模型中需要将这两个客户发送方与客户接收方模块实例化。记其实例的名称分别为 $q0$ 和 $q1$ 。客户发送方与客户接收方变量的说明如下。

$wbox[]$	长度为 win 的数组，取值范围 $\{0, 1, \dots, nbox - 1\}$ ，表示为 $p0$ 准备的等待传输的内容（表示为序号）。
cnt	取值范围 $\{0, \dots, win\}$ ，表示有效的 $wbox[]$ 内容的个数。
idx	取值范围 $\{0, 1, \dots, nbox - 1\}$ ，用以表示得到的信息的序号。
c	取值范围 $\{0, 1\}$ ，用以表示信息的内容，1 表示有内容，0 表示无内容。

客户发送方的变量 cnt 表示待发送的消息个数， $wbox[0], wbox[1]$ 表示需要发送的消息。初始准备的时候对 $wbox[0], wbox[1]$ 进行初始化赋值并赋 cnt 为 win 。然后 cnt 的值由 $p0$ 修改为该值减去发送成功的消息个数。若 $cnt < 2$ (win 为 2 的情况)，则客户发送方重新对 $wbox[0], wbox[1]$ 进行相应调整后，将 cnt 改成 2。对于 $p0$ 的迁移，将 $p0$ 的前两条迁移更改成如下迁移。

```

pc=s0&p1chanlen<win&q0.cnt=win:
(pc,p1chanidx[p1channext],p1chancont[p1channext],p1channext,p1chanlen):=
(s1,q0.wbox[k],1,(p1channext+1)%win,p1chanlen+1);
pc=s1&p0chanlen=1:
(pc,p0chanlen,x,k,q0.cnt):=
(s0,0,p0chanidx,0,q0.cnt-(((nbox+p0chanidx)-x)%nbox));

```

客户接收方的变量 idx 表示收到消息的序号, $c = 1$ 表示内容有效。这两个变量直接由 $p1$ 进行赋值。客户接收方的模块没有迁移动作。对于 $p1$ 的迁移, 将 $p1$ 的第4和第6两条迁移更改成如下迁移。

```

pc=r2&(box[y]=1):      (pc,box[y],y,q1.idx,q1.c):=(r3,0,(y+1)%nbox,y,c);
pc=r3&(box[y]=1):      (pc,box[y],y,q1.idx,q1.c):=(r3,0,(y+1)%nbox,y,c);

```

我们需要验证的性质为接收方按顺序接收到消息, 分成五条性质, 表示如下。

```

A(q1.c=0 U (q1.c=1&q1.idx=0));
AG(!(q1.c=1&q1.idx=0) | A((q1.c=1&q1.idx=0) U (q1.c=1&q1.idx=1)));
AG(!(q1.c=1&q1.idx=1) | A((q1.c=1&q1.idx=1) U (q1.c=1&q1.idx=2)));
AG(!(q1.c=1&q1.idx=2) | A((q1.c=1&q1.idx=2) U (q1.c=1&q1.idx=3)));
AG(!(q1.c=1&q1.idx=3) | A((q1.c=1&q1.idx=3) U (q1.c=1&q1.idx=0)));

```

这五条性质在可靠通道模型下成立, 在不可靠通道模型下不成立。但在不可靠通道模型下可以证明以下稍弱一些的相应性质。

```

A((q1.c=1&q1.idx=0) R ((q1.c=1&q1.idx=0) | q1.c=0));
AG(!(q1.c=1&q1.idx=0) | A((q1.c=1&q1.idx=1) R (q1.c=1&(q1.idx=0 | q1.idx=1))));
AG(!(q1.c=1&q1.idx=1) | A((q1.c=1&q1.idx=2) R (q1.c=1&(q1.idx=1 | q1.idx=2))));
AG(!(q1.c=1&q1.idx=2) | A((q1.c=1&q1.idx=3) R (q1.c=1&(q1.idx=2 | q1.idx=3))));
AG(!(q1.c=1&q1.idx=3) | A((q1.c=1&q1.idx=0) R (q1.c=1&(q1.idx=3 | q1.idx=0))));

```

若加上前面讨论的公平约束, 在不可靠通道模型下, 前五条性质成立。后五条的成立则可由前五条的成立得到。

对于 $q0$ 与 $p0$ 之间的消息传输和 $p1$ 与 $q1$ 之间的消息传输, 模型中用的是简单共享变量。若每对进程处于一台计算机内部, 那么共享变量是合理的。若每对进程处于具有可靠通道的不同计算机, 那么从建模角度, 则用通道(用多个共享变量实现, 有明确的存和取相关的迁移)比较合理。从模型规模考虑, 若用通道建模, 需要使用更多变量和迁移条数, 不便于自动验证。

§9.3 整数平方根算法

本节考虑用加法运算实现求一个数的整数平方根的算法。

整数平方根算法的卫式迁移模型： 给定 $B = (\{s_0, s_1, s_2, s_3, s_4, 0, 1, 2, +, *\}, \{=, \leq\})$ 。给定 $I = (Nat, I_0)$ 为 B 在自然数上的通常解释且 s_i 解释为 i 。

给定 $V = \{x, y_1, y_2, y_3, pc\}$ 。

(B, V) 上的整数平方根算法的卫式迁移模型定义为 $M = \langle T, \Theta \rangle$ 其中 Θ 为 $a = s_0$ 且 T 为以下迁移的集合：

$pc = s_0$	\rightarrow	$(y_1, y_2, y_3, pc) := (0, 1, 1, s_1)$
$pc = s_1 \wedge (y_3 \leq x)$	\rightarrow	$(pc) := (s_2)$
$pc = s_1 \wedge \neg(y_3 \leq x)$	\rightarrow	$(pc) := (s_4)$
$pc = s_2$	\rightarrow	$(y_1, y_2, pc) := (y_1 + 1, y_2 + 2, s_3)$
$pc = s_3$	\rightarrow	$(y_3, pc) := (y_3 + y_2, s_1)$

该模型状态迁移部分的图形表示见图3。 $pc = s_i$ 表示当前的计算位置处于 s_i 状态。

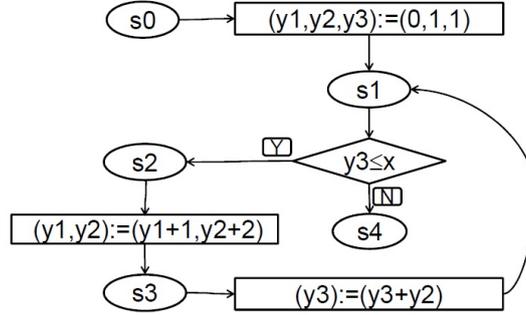


图3 整数平方根算法

整数平方根算法的性质： 我们考虑以下两个性质。

$\varphi_1 = (x > 0 \rightarrow \Box(pc = s_4 \rightarrow y_1 = \sqrt{x})).$ $\varphi_2 = (x > 0 \rightarrow \Diamond(pc = s_4)).$
--

其中 $a = \sqrt{b}$ 表示 $a^2 \leq b \wedge b < (a + 1)^2$ 。这两个PLTL性质亦可用CTL表示如下。

$\varphi'_1 = (x > 0 \rightarrow AG(pc = s_4 \rightarrow y_1 = \sqrt{x})).$ $\varphi'_2 = (x > 0 \rightarrow AF(pc = s_4)).$
--

§9.3.1 演绎推理

(1) $M \models_I \varphi_1$ 的证明如下。

应用§7.1节的证明结论形式为 $\varphi \Rightarrow \Box\psi$ 的证明规则， $M \models_I \varphi_1$ 可根据以下步骤证明。

首先证明 $M \models_I \Box(\Theta \wedge x > 0 \rightarrow \Box(pc = s_4 \rightarrow y_1 = \sqrt{x}))$ 如下。

定义 $\varphi = (\Theta \wedge x > 0)$ 且 $\psi = (pc = s_4 \rightarrow y_1 = \sqrt{x})$ 。

定义 ϕ_0, \dots, ϕ_4 如下。

$$\begin{aligned}\phi_0 &= (x > 0) \\ \phi_1 &= (y_1^2 \leq x \wedge y_2 = 2 * y_1 + 1 \wedge y_3 = (y_1 + 1)^2) \\ \phi_2 &= ((y_1 + 1)^2 \leq x \wedge y_2 = 2 * y_1 + 1 \wedge y_3 = (y_1 + 1)^2) \\ \phi_3 &= (y_1^2 \leq x \wedge y_2 = 2 * y_1 + 1 \wedge y_3 = y_1^2) \\ \phi_4 &= (y_1 = \sqrt{x})\end{aligned}$$

定义 ϕ 为 $\bigvee_{i=0}^4 (pc = s_i \wedge \phi_i)$ 。

则有 $\varphi \Rightarrow \phi$ 且 $\phi \rightarrow [T]\phi$ 且 $\phi \Rightarrow \psi$ 。

因而根据证明规则我们有 $\varphi \Rightarrow \Box\psi$ ，即 $M \models_I \Box(\Theta \wedge x > 0 \rightarrow \Box\psi)$ 。

因而 $M \models_I \Theta \wedge x > 0 \rightarrow \Box\psi$ 。进而我们有 $M \models_I \varphi_1$ 。

(2) $M \models_I \varphi_2$ 的证明如下。

应用§7.1节的结论形式为 $\varphi \Rightarrow \Diamond\psi$ 的证明规则， $M \models_I \varphi_3$ 可根据以下步骤证明。

首先证明 $M \models_I \Box((pc = s_0 \wedge x > 0) \rightarrow \Diamond(pc = s_4))$ 如下。

扩展 B 使得 $B' = (F \cup \{f\}, P)$ 。扩展 I_0 使得以下成立。

$$\begin{aligned}I(f(s_0, x, y_3)) &= 3x + 1 \\ I(f(s_i, x, y_3)) &= 3(x + 1 - y_3) + 1 - i \quad (i = 1, 2, 3) \\ I(f(s_4, x, y_3)) &= 0\end{aligned}$$

定义 $\varphi, \psi, w, e, \phi$ 如下。

$$\begin{aligned}\varphi &= (pc = s_0 \wedge x > 0) \\ \psi &= (pc = s_4) \\ w &= true \\ e &= f(pc, x, y_3) \\ \phi &= \bigvee_{i=0}^3 (pc = s_i \wedge \phi_i)\end{aligned}$$

易知 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, I(\leq)) = (Nat, \leq)$ 为良基集合。

我们有 $\varphi \Rightarrow (\psi \vee \phi)$ ， $\phi \Rightarrow w_x^e \wedge (\psi \vee E(T))$ 且 $\phi \wedge e = v \rightarrow [T](\psi \vee (\phi \wedge e < v))$ 。

因而根据证明规则我们有 $\varphi \Rightarrow \Diamond\psi$ ，即 $M \models_I \Box((pc = s_0 \wedge x > 0) \rightarrow \Diamond(pc = s_4))$ 。

因而 $M \models_I (pc = s_0 \wedge x > 0) \rightarrow \Diamond(pc = s_4)$ 。进而我们有 $M \models_I \varphi_2$ 。

(3) 若 B 改在整数论域上解释，则在验证 $M \models_I \varphi_2$ 时，各元素的定义和选取会略有不同。首先可修改 f 的定义如下。

$$\begin{aligned}I(f(s_0, x, y_2, y_3)) &= 3x + 4 \\ I(f(s_i, x, y_2, y_3)) &= 3(x + 1 + y_2 - y_3) + 1 - i \quad (i = 1, 2) \\ I(f(s_3, x, y_2, y_3)) &= 3(x + 1 + y_2 - y_3) - 8 \\ I(f(s_4, x, y_2, y_3)) &= 0\end{aligned}$$

然后可选取 $\varphi, \psi, w, e, \phi$ 如下。

$\varphi = (pc = s_0 \wedge x > 0)$	$\phi = (pc = s_0 \wedge \phi_0) \wedge$
$\psi = (pc = s_4)$	$(pc = s_1 \wedge \phi_1 \wedge y_2 \geq 1) \wedge$
$w = (x \geq 0)$	$(pc = s_2 \wedge \phi_2 \wedge y_2 \geq 1) \wedge$
$e = f(pc, x, y_2, y_3)$	$(pc = s_3 \wedge \phi_3 \wedge y_2 \geq 3)$

易知 $W = (\{\sigma(x) \mid I(w)(\sigma) = true\}, I(\leq)) = (Nat, \leq)$ 为良基集合。其余部分的证明类似自然数解释下的证明。

§9.3.2 基于模型检测工具的自动验证

由于模型检测算法主要是基于穷尽搜索，我们需要对变量的取值范围进行限制。

自动验证： 设变量的值域范围为 $\{0, 1, \dots, 30\}$ 。应用VERDS建模语言VML建立VERDS验证模型如下。

```

/*****/
VVM      isqrt1.vvm
VAR      pc:{s0,s1,s2,s3,s4}; x:0..30; y1:0..30; y2:0..30; y3:0..30;
INIT     pc=s0; x<=30; y1=0; y2=0; y3=0;
TRANS    pc=s0:                (y1,y2,y3,pc):=(0,1,1,s1);
         pc=s1&(y3<=x):        (pc):=(s2);
         pc=s1&!(y3<=x):       (pc):=(s4);
         pc=s2:                (y1,y2,pc):=(y1+1,y2+2,s3);
         pc=s3:                (y3,pc):=(y3+y2,s1);
SPEC     !(x>0) | AG(!(pc=s4) | (x>=y1*y1) & x<(y1+1)*(y1+1));
         !(x>0) | AF(pc=s4);
/*****/

```

应用VERDS验证 φ'_1, φ'_2 的结果说明这些性质（在变量取值范围受限的情况下）成立。

§9.3.3 错误检查与模型检测中的反例生成

假定我们在模型中把条件 $y_3 \leq x$ 误写为 $y_3 < x$ 。应用VERDS建模语言VML建立修改的VERDS验证模型（即将两处 $y_3 \leq x$ 改为 $y_3 < x$ ）并验证 φ'_1, φ'_2 ，则有以下结果： φ'_2 成立而 φ'_1 不成立。验证结果说明这个错误影响计算结果的正确性，但不影响模型运行的终止性。

反例生成： 对前面不成立的性质VERDS可提供反例，即不满足性的见证。整理后可得如下不满足 φ'_1 的运行路径。

状态	pc	x	y1	y2	y3
0	s0	1	0	0	0
1	s1	1	0	1	1
2	s4	1	0	1	1

这条运行路径显示开始时 x 的值为1，最后 $pc = s4$ 时 $y1$ 的值为0，而此时的 $y1$ 值应为1。因而模型不满足 $x > 0 \rightarrow AG((x \geq y1 * y1) \wedge x < (y1 + 1) * (y1 + 1))$ 。若改为验证 $x > 5 \rightarrow AG((x \geq y1 * y1) \wedge x < (y1 + 1) * (y1 + 1))$ ，那么使得性质不满足的最小的 x 的值为9。模型检测工具VERDS所提供的反例路径如下。

状态	pc	x	y1	y2	y3
0	s0	9	0	0	0
1	s1	9	0	1	1
2	s2	9	0	1	1
3	s3	9	1	3	1
4	s1	9	1	3	4
5	s2	9	1	3	4
6	s3	9	2	5	4
7	s1	9	2	5	9
8	s4	9	2	5	9

这条运行路径显示开始时 x 的值为9, 最后 $pc = s4$ 时 $y1$ 的值为2, 而此时的 $y1$ 值应为3。因而模型不满足 $x > 5 \rightarrow AG((x \geq y1 * y1) \wedge x < (y1 + 1) * (y1 + 1))$ 。

§9.4 模型中特定状态和路径的查找

特定状态： 特定状态的查找问题就是问是否存在可达的满足给定性质的状态。

设给定性质表示为状态公式 φ , 则该问题可转化为检查模型是否满足 $AG(\neg\varphi)$ 。若所检查的性质不满足, 则通过查看反例可获得满足 φ 的一个可达状态, 以及模型运行是怎么到达这个状态的。

模型检测可在很大状态空间中查找具有特定性质的可达状态。应用模型检测工具VERDS, 对于一些简单的模型和性质, 搜索的可达状态空间可以达到 10^{50} 个状态。

以窗口协议为例。我们可以问 $x = 1$ 且 $y = 3$ 的状态是否可达。我们验证 $AG\neg(x = 1 \wedge y = 3)$ 是否成立。如果成立则不可达, 如果不成立则可达。§9.2.1节的可靠通道模型不满足 $AG\neg(x = 1 \wedge y = 3)$ 。模型检测工具VERDS所提供的一条到达 $x = 1$ 且 $y = 3$ 的状态的路径 π 的形式为 $\pi_0\pi_1 \cdots \pi_{19} \cdots$, 其中路径部分状态上的部分变量的值如下。

状态	p0.pc	p1.pc	p0.x	p1.y
0	s0	r0	0	0
4	s1	r3	0	1
7	s0	r0	1	1
10	s1	r3	1	2
19	s1	r3	1	3

特定路径： 特定路径的查找问题就是问是否存在计算路径满足给定的性质。

若给定性质表示为 $\diamond(\varphi)$ 其中 φ 为状态公式, 则该问题亦可转化为检查模型是否满足 $AG(\neg\varphi)$ 。若所检查的性质不满足, 则通过查看反例可获得一条满足 $\diamond(\varphi)$ 的路径。

若给定性质表示为 $\square(\varphi)$ 其中 φ 为状态公式, 则该问题亦可转化为检查模型是否满足 $AF(\neg\varphi)$ 。若所检查的性质不满足, 则通过查看反例可获得一条满足 $\square(\varphi)$ 的路径。

通常情况下, 路径由于可以任意长, 路径空间是无穷的, 那么其中可能有一小部分路径满足给定性质。模型检测工具可以应用于搜索小概率的满足一定性质的路径。

以窗口协议为例。我们可以问是否存在路径上要么 x 不是1要么 y 不是3的计算。那么我们验证 $AF(x = 1 \wedge y = 3)$ 是否成立。如果成立，则不存在这样的计算，如果不成立则存在。§9.2.1节的可靠通道模型不满足 $AF(x = 1 \wedge y = 3)$ 。模型检测工具VERDS所提供的无穷计算路径显示这条路径 π 的形式为 $\pi_0\pi_1(\pi_2 \cdots \pi_{29})^\omega$ ，其中路径部分状态上的部分变量的值如下。

状态	p0.pc	p1.pc	p0.x	p1.y	p1.n
0	s0	r0	0	0	0
2	s1	r1	0	0	0
4	s1	r3	0	1	0
7	s0	r0	1	1	0
9	s1	r1	1	1	1
11	s1	r3	1	2	1
14	s0	r0	2	2	1
16	s1	r1	2	2	2
18	s1	r3	2	3	2
21	s0	r0	3	3	2
23	s1	r1	3	3	3
25	s1	r3	3	0	3
28	s0	r0	0	0	3

模型中区分 π_{28} 和 π_0 的仅仅是 n 和 c 的值。若在模型的构造中选择将 n, c 在使用后重置为0，那么不满足 $F(x = 1 \wedge y = 3)$ 的路径的形式为 $(\pi'_0\pi'_1 \cdots \pi'_{27})^\omega$ ，其中 n, c 的值在部分非关键的对应状态 π'_i 与 π_i 上略有不同。

§9.5 练习

1. 设计基于加减法运算的 n 模 k 算法（输入为 n 和 k ）。应用推理方法证明该算法正确性。将变量取值范围进行适当限制，用模型检测工具验证受限的算法的正确性。
2. 设计3个进程的环选举算法并构造迁移模型。证明（或应用工具辅助证明）该算法满足安全性质（每一个进程要么不知道结果要么知道正确的结果）和必达性质（选举算法总会有结果且每一个进程都知道结果）。

附录 A 建模语言VML

本附录介绍基于卫式迁移模型的建模语言VML。
用VML写的模型称为VVM模型，即VERDS验证模型。

1. Atoms

An atom may be any sequence of characters starting with a character in the set {a-z_} and followed by a possibly empty sequence of characters belonging to the set {a-z0-9_}. A number is any sequence of digits. A digit belongs to the set {0-9}.

All characters and case in a name are significant. Whitespace characters are space (SPACE), tab (TAB) and newline (RET). Comments in verds language is any string starting with two slashes ('//') and ending with a newline. Any other tokens recognized by the parser are enclosed in quotes in the syntax expressions below. Grammar productions enclosed in square brackets ('[]') are optional.

2. Expressions

Expressions are constructed from variables, constants, and a collection of operators, including temporal operators, Boolean connectives, and integer arithmetic operators.

2.1. Numeric Expressions

Numeric expressions are expressions built only from current state variables. The syntax of numeric expressions is as follows:

```
numeric_expr ::
    symb_const          ;; a symbolic constant
  | numb_const         ;; a numeric constant
  | variable_id        ;; a variable identifier
  | "(" numeric_expr ")"
  | numeric_expr "+" numeric_expr    ;; integer addition
  | numeric_expr "-" numeric_expr    ;; integer subtraction
  | numeric_expr "*" numeric_expr    ;; integer multiplication
  | numeric_expr "/" numeric_expr    ;; integer division
  | numeric_expr "%" numeric_expr    ;; integer remainder
```

```
symb_const :: atom
```

```
numb_const :: number
```

```
variable_id :: atom | atom"."atom | variable_id "[" numeric _expr "]"
```

A symbolic constant is represented by an atom, which are predefined ones, including the atom "pid". Such constants are meant to be used in particular sections of the declarations.

A numeric constant is represented by a number. A simple variable identifier is either an atom representing a simple variable or such a variable identifier preceded by a process identifier (represented by the first atom in the construct atom"."atom) in order to refer a local variable of a process. A simple variable identifier followed by a sequence of indices represents an element of an array variable.

The order of parsing precedence for operators from high to low is:

```
*,/,%
```

```
+,-
```

Operators of equal precedence associate to the left. Parentheses may be used to group expressions.

2.2. Logic Expressions

Logic expressions are expressions built from numeric expressions. Logic expressions can be used to specify sets of states, e.g. the initial set of states. The syntax of logic expressions is as follows:

```
logic_expr ::
    "TRUE"                ;; The boolean constant 1
  | "FALSE"               ;; The boolean constant 0
  | logic_const           ;; predefined constants
  | "(" logic_expr ")"
  | numeric_expr "=" numeric_expr    ;; equality
  | numeric_expr "!=" numeric_expr   ;; inequality
  | numeric_expr "<" numeric_expr     ;; less than
```

```

| numeric_expr ">" numeric_expr      ;; greater than
| numeric_expr "<=" numeric_expr     ;; less than or equal
| numeric_expr ">=" numeric_expr     ;; greater than or equal
| logic_expr "&" logic_expr          ;; logical and
| logic_expr "|" logic_expr           ;; logical or
| "!" logic_expr                       ;; logical not

```

```
logic_const :: atom
```

A logical constant is represented by an atom, which are predefined ones, including the atom "running". Such constants are meant to be used in particular sections of the declarations.

The order of parsing precedence for operators from high to low is:

```

=, !=, <, >, <=, >=
!
&
|

```

Operators of equal precedence associate to the left. Parentheses may be used to group expressions.

2.3. Temporal Logic Expressions

Temporal logic expressions are expressions built from logic expressions. Logic expressions can be used to specify properties of the finite state machine. The syntax of temporal logic expressions is as follows:

```

temporal_expr ::
    logic_expr
  | "AX" temporal_expr
  | "AG" temporal_expr
  | "AF" temporal_expr
  | "A" "(" temporal_expr "R" temporal_expr ")"
  | "A" "(" temporal_expr "U" temporal_expr ")"
  | "EX" temporal_expr
  | "EG" temporal_expr
  | "EF" temporal_expr
  | "E" "(" temporal_expr "R" temporal_expr ")"

```

```

| "E" "(" temporal_expr "U" temporal_expr ")"
| temporal_expr "&" temporal_expr          ;; logical and
| temporal_expr "|" temporal_expr          ;; logical or
| "!" temporal_expr                          ;; logical not

```

3. Discrete Transition Systems

A discrete transition system consists three parts: the variables representing the state of the system, the initial states, and the transition relation. In addition, property specifications may be attached to a discrete transition system, in order to check whether the discrete transition system is correct with respect to the properties. The transition relation may be divided into different modules.

3.1. State Variables

A state of the model is an assignment of values to a set of state variables. These variables are declared by the notation:

```
var_declaration :: variable ":" type ";" ... variable ":" type ";"
```

```
variable :: atom | variable "[" number ".." number "]"
```

```
type :: number ".." number | "{" atom "," atom "," ... "," atom "}"
```

A variable is either a simple variable or an array variable. The type associated with a variable declaration can be either a scalar or an enumeration of a set of atoms.

Two enumeration types must either be identical or with disjoint sets of elements.

A variable of type Boolean may be represented by a variable of type 0..1 such that $x=1$ means x for the Boolean variable x , $x=0$ means not x , and negating the Boolean variable x may be implemented by the arithmetic expression $1-x$.

3.2. Initial Values of State Variables

Initial values of state variables are specified by a list of formulas

constraining the state variables. As a special case, the initial values of state variables may be specified as a list of formulas of the form $x=c$ where x is a variable identifier and c is a constant. The initial values are declared by the notation:

```
init_declaration :: logic_expr ";" ... logic_expr ";"
```

3.3. Transition Relations

The transition relation of a process is implemented by a set of transition relations. The transition relation is declared by the notation:

```
trans_declaration :: cond ":" assignment ";" ... cond ":" assignment ";"
```

```
cond :: logic_expr
```

```
assignment ::
```

```
  "(" variable_id "," variable_id "," ... variable_id ")"
```

```
  " :="
```

```
  "(" numeric_expr "," numeric_expr "," ... numeric_expr ")" ";"
```

3.4. Fairness Declarations

The transition relation may be augmented with fairness requirements that impose restrictions on the valid execution paths. The specification is declared by the notation:

```
fairness_declaration :: logic_expr ";" ... logic_expr ";"
```

The logical constant "running" is a special proposition that may be used in a fairness declaration. The proposition is satisfied when the process in which the fairness is declared is executed.

3.5. Specification Declarations

The specification of system and process properties is represented by a temporal logic expression. The specification is declared by the notation:

```
spec_declaration :: temporal_expr ";" ... temporal_expr ";"
```

3.6. Module Declarations

A module is an encapsulated collection of declarations. Once defined, a module can be reused as many times as necessary. Modules can also be so that each instance of a module can refer to different data values.

The syntax of a module declaration is as follows.

```
module ::
    "MODULE"      atom "(" [ par_list ] ")"
    "VAR"         [ var_declaration ]
    "INIT"        init_declaration
    "TRANS"       trans_declaration
    [ "FAIRNESS"  fairness_declaration ]
```

```
par_list :: atom | atom "[" | par_list "," par_list
```

The atom immediately following the keyword "MODULE" is the name associated with the module. The optional par_list in parentheses is the list of the formal parameters of the module. The symbols "[" indicates that the atom preceding the symbols represents an array variable. Whenever these parameters occur in expressions within the module, they are replaced by the actual parameters which are supplied when the module is instantiated.

The symbolic constant "pid" is a special constant that may be used in a module declaration. The constant is interpreted as the numerical constant representing the id of the process (the first process has id=0), which is an instance of the module.

3.7. Transition system declarations

A discrete transition system may have one or more processes that are instances of one or more module declarations. The syntax of a discrete transition system (called verds verification model) declaration is as follows.

```
verds_verification_model ::
    "VVM"         [ atom ]
    [ "DEFINE"    definition_declaration ]
    "VAR"         [ var_declaration ]
```

```

"INIT"          init_declaration
"PROC"          process_declaration
[ "FAIRNESS"    fairness_declaration ]
[ "SPEC"        spec_declaration ]

```

```
definition_declaration ::
```

```

string_latin_letters "=" string_visible_characters "\n"
...
string_latin_letters "=" string_visible_characters "\n"

```

```
process_declaration ::
```

```

atom ":" atom "(" [ par_list ] ")" ";"
...
atom ":" atom "(" [ par_list ] ")" ";"

```

The optional atom after the keyword VVM is the name of the discrete transition system.

A list of definitions may be provided. A defined word (which is a string of Latin letters) is an abbreviation of the defining string of visible characters, and the end-line symbol indicates the end of the definition. If a defining string is long, a backslash '\' may be used to ignore the end-line character immediately after this backslash.

A process is an instance of a module. The first atom in the process declaration is the name of the process (or the module instance), the second atom is the name of a module and the atoms in the optional list are parameters passed to the module.

The variables in the var_declaration section are global variables that can be used in all modules.

The complete verification model consists of a verds_verification_model declaration followed by a list of module declarations that are referred to in the process declaration.

For simplicity, a single process discrete transition system may be declared by the following syntax as well.

```
single_process_verds_verification_model ::
```

```
"VVM"          [ atom ]
"VAR"          [ var_declaration ]
"INIT"         init_declaration
"TRANS"        trans_declaration
[ "FAIRNESS"   fairness_declaration ]
[ "SPEC"       spec_declaration ]
```

This declaration is similar to the previous declaration of a VVM with the difference that the process declaration is replaced by a transition declaration.

附录 B 参考文献

1. Alur, R., Dill, D.L.: A theory of timed automata. *Theor. Comput. Sci.* 126(2), 183-235 (1994)
2. Ben-Ari, M., Manna, Z., Pnueli, A.: The Temporal Logic of Branching Time. In: *POPL'81*. pp. 164-176 (1981)
3. Bhat, G., Cleaveland, R.: Efficient Local Model-Checking for Fragments of the Modal μ -Calculus. In: *TACAS'96*. pp. 107-126 (1996)
4. Biere, A., Cimatti, A., Clarke, E., Strichman, O., , Zhu, Y.: Bounded model checking. *Advances in Computers* 58, 117-148 (2003)
5. Biere, A., Cimatti, A., Clarke, E.M., Zhu, Y.: Symbolic Model Checking without BDDs. In: *TACAS'99*. pp. 193-207 (1999)
6. Bradfield, J., Esparza, J., Mader, A.: An Effective Tableau System for the Linear Time μ -Calculus. In: *ICALP'97*. pp. 98-109 (1997)
7. Buccafurri, F., Eiter, T., Gottlob, G., Leone, N.: On actl formulas having linear counterexamples. *J. Comput. Syst. Sci.* 62, 463-515 (2001)
8. Burch, J.R., Clarke, E.M., McMillan, K.L., Dill, D.L., Hwang, J.: Symbolic model checking: 10^{20} states and beyond. In: *LICS'90*. pp. 428-439 (1990)
9. Clark, E.M., Grumberg, O., Peled, D.: *Model Checking*. MIT press (1999)
10. Clarke, E.M., Emerson, E.A.: Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In: *Logic of Programs 1981*. pp. 52-71 (1981)
11. Courcoubetis, C., Vardi, M., Wolper, P., Yannakakis, M.: Memory-efficient algorithms for the verification of temporal properties. *Formal Methods in System Design* 1(2-3), 275-288 (1992)
12. Dax, C., Hofmann, M., Lange, M.: A Proof System for the Linear Time μ -Calculus. In: *FSTTCS'06*. pp. 98-109 (2006)
13. Demri, S., Schnoebelen, P.: Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In: *STACS'98*. pp. 61-72 (1998)
14. Emerson, E.A.: Formal models and semantics. *Temporal and Modal Logic* pp. 995-1072 (1990)
15. Emerson, E.A., Halpern, J.Y.: "sometimes" and "not never" revisited: on branching versus linear time temporal logic. *J. ACM* 33(1), 151-178 (1986)
16. Francez, N.: *Program verification*. Addison-Wesley Publishing Company Inc. (1992)
17. Gabbay, D.M., Pnueli, A., Shelah, S., Stavi, J.: On the temporal analysis of fairness. In: *POPL'80*. pp. 163-173 (1980)

18. Henzinger, T.A.: The Theory of Hybrid Automata. In: LICS'96. pp. 278-292 (1996)
19. Hoare, C.A.R.: An axiomatic basis for computer programming. *Commun. ACM* 12, 576-580 (1969)
20. Holzmann, G.J.: Design and Validation of Computer Protocols. Prentice Hall (1990)
21. Holzmann, G.J.: The SPIN Model Checker. Addison-Wesley (2003)
22. Kaivola, R.: A Proof System for the Linear Time μ -Calculus. In: CONCUR'95. pp. 423-437 (1995)
23. Kozen, D.: Results on the propositional μ -calculus. *Theor. Comput. Sci.* 27, 333-354 (1983)
24. Loeckx, J., Sieber, K.: The foundation of program verification. John Wiley & Sons Ltd. (1984)
25. Maidl, M.: The common fragment of CTL and LTL. In: FOCS'00. pp. 643-652 (2000)
26. Manna, Z., Pnueli, A.: How to cool a temporal proof system for your pet language. In: POPL'83. pp. 141-154 (1983)
27. Manna, Z., Pnueli, A.: Completing the temporal picture. *Theor. Comput. Sci.* 83, 97-130 (1991)
28. McMillan, K.L.: Symbolic Model Checking. Kluwer Publisher (1993)
29. Nielsen, M., Winskel, G.: Petri nets and bisimulation. *Theor. Comput. Sci.* 153(1- 2), 211-244 (1996)
30. Peled, D.A.: Software Reliability Methods. Springer (2001)
31. Pnueli, A.: The Temporal Logic of Programs. In: FOCS'77. pp. 46-57 (1977)
32. Ramsey, F.P.: On a problem of formal logic. *Proceedings of the London Mathematical Society* 30(1), 264-286 (1930)
33. Reynolds, M.: An axiomatization of full computation tree logic. *The Journal of Symbolic Logic* 66(3), 1011-1057 (2001)
34. Schneider, K.: Verification of reactive systems: formal methods and algorithms. Springer (2004)
35. Schnoebelen, P.: The complexity of temporal logic model checking. *Advances in Modal Logic* 4, 1-44 (2002)
36. Sistla, A.P., Clarke, E.M.: The complexity of propositional linear temporal logics. *J. ACM* 32(3), 733-749 (1985)
37. Tarjan, R.: Depth first search and linear graph algorithms. *SIAM Journal on Computing* 1(2), 146-160 (1972)
38. Vardi, M.Y.: A Temporal Fixpoint Calculus. In: POPL'88. pp. 250-259. ACM (1988)

39. Vardi, M.Y., Wolper, P.: An automata-theoretic approach to automatic program verification. In: LICS'86. pp. 332-344 (1986)
40. Zhang, W.: Bounded Semantics of CTL and SAT-based Verification. In: ICFEM'09. pp. 286-305 (2009)
41. Zhang, W.: QBF Encoding of Temporal Properties and QBF-Based Verification. In: IJCAR'14. pp. 286-305 (2014)
42. Zhang, W.: Bounded semantics. *Theor. Comput. Sci.* 564, 1-29 (2015)