概率时态认知逻辑模型检测中三值抽象技术的研究

周从华,孙 博,刘志锋,葛 云
(1.江苏大学计算机科学与通信工程学院,镇江,212013;2.南京大学电子科学与工程学院,南京,210093)

摘 要: 为缓解概率时态认知逻辑模型检测中的状态空间爆炸问题,提出了概率时态认知逻辑的三值抽象技术.具体研究内容包括:定义抽象模型及模型上概率时态认知逻辑的三值语义,依据状态空间等价划分建立初始抽象模型,并证明抽象技术对概率时态认知逻辑的满足性保持关系;提出概率时态认知逻辑模型检测算法;依据初始模型检测的结果,给出利用最小证据和最小反例引导的抽象系统的求精过程.最后通过 Dining Cryptographer 协议说明了抽象技术的应用,及其在约简系统状态空间方面的效果.

关键词: 三值抽象; 模型检测; 概率时态认知逻辑; 反例
中图分类号: TP301 **文献标识码:** A **文章编号:** 0372-2112 (2012)10-2052-10
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2012.10.025

Three-Valued Abstraction for Model Checking the Probabilistic Temporal Logic of Knowledge

ZHOU Cong-hua, SUN Bo, LIU Zhi-feng, GE Yun
(1. *School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang* 212013, *China*
2. *School of Electronic Science and Engineering, Nanjing University, Nanjing* 210093, *China*)

Abstract: In order to overcome the state explosion problem in model checking the probabilistic temporal logic of knowledge, a three-valued abstraction is proposed. Our work includes three parts: first the three-value semantics of the probabilistic temporal logic of knowledge is defined on the abstract model, and the initial abstract model is build according to the equivalence partition of state space, and the preservation of satisfaction under the abstraction is proved; second the model checking algorithm of the probabilistic temporal logic of knowledge is proposed; third how to refine the abstraction by the minimal witnesses and counterexamples generated in model checking is shown. Finally, the abstraction is applied in model checking the Dining Cryptographer protocols.

Key words: three-valued abstraction; model checking; probabilistic temporal logic of knowledge; counterexample

1 引言

模型检测[1,2]是一种有限状态系统自动化验证技术,已经在计算机硬件、安全认证协议等方面获得了广泛的应用.概率时态认知逻辑[3,4]作为一种属性规范描述语言,可以对随机算法,通信协议,复杂系统(比如分布式和多智体系统)的规范进行描述.借助于模型检测方法判断这些系统是否满足概率时态认知逻辑描述的属性规范,对确保系统的可信性具有重要意义.

和传统模型检测技术一样,概率时态认知逻辑模型检测的主要挑战是状态空间爆炸问题,即状态空间随着并发分量的增加呈指数级增长.为克服该问题,研究人员提出了一系列状态空间简化技术,主要包括基于 OBDD 的符号化计算[5]、偏序归约[6]、对称归约[7]、抽象[8,9]等,其中抽象技术是克服空间爆炸最为有效的方法之一.抽象通过删除原始模型中与验证无关的信息得到一个抽象模型,属性验证在抽象模型中进行.由于状态空间相对较小,验证效率大为提高.

传统的二值抽象技术框架下,抽象模型是原始模型的上近似,即如果抽象模型满足属性,则原始模型也满足属性.但是当抽象模型不满足待验证属性时,不能演绎出原始系统也不满足属性.三值抽象技术[10]通过引入除真和假之外表示不确定性的第三值有效地克服了这一不足.三值抽象技术的研究主要包括抽象模型的建立和求精[11]两个方面.目前,时态逻辑模型检测中这两个方面均已得到深入研究,概率时态逻辑模型检测中抽

收稿日期:2010-12-23;修回日期:2011-05-22
基金项目:国家自然科学基金(No.61003288,No.6111130184);江苏省自然科学基金(No.BK2010192);教育部博士点基金(No.20093227110005)

象模型的建立亦已有相应的方法[12],但是抽象求精目前还没有相关的工作进行讨论,而在概率时态认知逻辑的模型检测中,就我们所知,目前这两个方面均没有得到任何的探讨.因此本文的主要工作是系统研究模型检测概率时态认知逻辑中的三值抽象技术,具体工作包括三个方面:

1) 三值抽象:定义抽象模型,该模型的主要特点是利用概率转换区间替换了原始模型中的概率转换,概率转换的上界和下界分别对应上近似和下近似抽象;在抽象模型上定义概率时态认知逻辑的三值语义,并建立逻辑公式在抽象模型上的满足性和在原始模型上的满足性之间的关系.进一步研究如何从状态空间的等价划分演绎抽象模型.

2) 模型检测算法:开发在抽象状态空间上检测概率时态认知逻辑的算法.

3) 抽象求精:分析抽象失败的原因,依据模型检测的结果计算出最小证据集和最小反例集,开发利用最小证据集和最小反例集引导抽象系统求精的算法,从而确保抽象技术的完备性.

为测试三值抽象技术在缓解状态空间方面的效果,开发了专门验证 Dining Cryptographer[13,14] 协议的模型检测工具 DCcheck.实验结果表明抽象技术可以降低内存需求大约 30%,降低验证时间 25%.

2 概率时态认知逻辑及其语义

目前出现了很多组合时态、认知、概率的逻辑系统,例如 P_rKD45, *PETL*,这些逻辑系统中概率因素仅仅被用来对认知进行不确定推理,一个自然的扩展就是利用概率对系统动态行为的不确定性进行推理,如"系统成功的概率是 99%".本小节我们定义一种概率时态认知逻辑 *PTLK* 来表示多智体系统中与系统动态行为和认知相关的概率行为.设 $Ap=\{a,b,\ldots\}$ 为有限原子命题集,$Ag=\{1,\ldots,n\}$ 表示由 n 个智体组成的多智体系统[15],其中 $i\in Ag$ 表示第 i 个智体.

定义 1 (*PTLK* 语法) Ap 和 Ag 上的 *PTLK* 递归定义如下:

- 如果 $a\in Ap$,则 a 是 *PTLK* 公式;
- 如果 φ,φ 是 *PTLK* 公式,则 $\neg\varphi,\varphi\wedge\varphi,\varphi\vee\varphi$ 是 *PTLK* 公式;
- 如果 φ,φ 是 *PTLK* 公式,则 $X^{\triangleright p}\varphi,\varphi U^{\triangleright p}\varphi$ 是 *PTLK* 公式,这里 X,U 分别是线性时态逻辑 *LTL*[1]中表示"*next time*"和"*until*"的时态算子,$\triangleright\in\{<,\leqslant,>,\geqslant\}$,$p\in[0,1]$为实数;
- 如果 φ 是 *PTLK* 公式,则 $K_i^{\triangleright p}\varphi$ 是 *PTLK* 公式,这里 K 是认知逻辑[15]中表示"知道"的知识算子,$\triangleright\in\{<,\leqslant,>,\geqslant\}$,$p\in[0,1]$,$i\in Ag$.

为了能够描述随机现象,我们需要对状态转换系统进行扩展,以便来解释 *PTLK* 的语义.

定义 2 (*PTLK* 语义模型) Ap 和 Ag 上的概率 *Kripke* 结构 M 是一个多元组 $(S,s_0,\sim_1,\ldots,\sim_n,P,P_1,\ldots,P_n,L)$,这里

- S 是有限状态集;
- $s_0\in S$ 是初始状态;
- $\sim_i\subseteq S\times S(1\leqslant i\leqslant n)$是认知关系:$s_1\sim_i s_2$ 当且仅当状态 s_1,s_2 对智体 i 是不可区分的,即在 s_1,s_2 下智体 i 的局部状态是相同的;
- $P:S\times S\to[0,1]$是状态转换概率函数,满足对任意的 $s\in S$,$\sum_{s'\in S}P(s,s')=1$;
- $P_i:S\times S\to[0,1](1\leqslant i\leqslant n)$是认知关系上的概率函数,满足对任意的 $s\in S$,$\sum_{s'\in S}P_i(s,s')=1$;
- $L:S\to 2^{Ap}$是状态标记函数,为每个状态指定该状态下值为真的命题.

在定义 *PTLK* 的语义之前,我们首先回顾一下概率论方面的基本内容.一项随机试验中所有可能发生的结果形成的集合称为样本空间,记为 Ω.集合 $\Pi\subseteq 2^{\Omega}$ 称为 Ω 上的 σ 代数,如果 Π 包含 Ω,$\Omega\setminus E(E\in\Pi)$,以及 Π 中任何可数元素的并集.Π 中任何元素是可度量的.概率空间是一个三元组 $PS=(\Omega,\Pi,Prob)$,这里 Ω 为样本空间,集合 Π 为 Ω 上的 σ 代数,$Prob:\Pi\to[0,1]$是度量函数,满足 $Prob(\Omega)=1$,且对 Π 两两不相交的序列 $E_1,E_2,\ldots$,$Prob(\bigcup_{i=1}^{\infty}E_i)=\bigcup_{i=1}^{\infty}Prob(E_i)$.

定义 3 (路径)设 M 为概率 *Kripke* 结构.

- M 中的无穷路径是一个无穷状态序列 $s_1s_2\ldots$ 使得 $\forall\, i\geqslant 1,P(s_i,s_{i+1})>0$.
- 无穷路径的有限前缀称为有穷路径.

对于概率 *Kripke* 结构 M,我们可以演绎出如下的概率空间:$\Omega\subseteq S^{\omega}$ 为 M 中无穷路径的集合.Π 为 σ 代数定义为 $\Pi=\{C(\pi)\mid\pi\in S^{*}\}$,这里 $C(\pi)=\{\pi'\in\Omega\mid\pi\text{ 是 }\pi'\text{ 的有限前缀}\}$,$\Pi$ 上的概率度量 *Prob* 定义为 $Prob(C(s_1s_2\ldots s_n))=\prod_{1\leqslant i\leqslant n-1}P(s_i,s_{i+1})$.令记号 $Path_M^{\omega}(s)$表示 M 中从状态 s 出发的路径集合,记号 $Path_M^{\omega}(s,\varphi)$表示 M 中从状态 s 出发的满足 φ 的路径集合.形式化的来讲,$Path_M^{\omega}(s,\varphi)=\{\pi\in Path_M^{\omega}(s)\mid\pi\text{ 满足 }\varphi\}$.定义 $Prob(s,\varphi U\varphi)=Prob(Path_M^{\omega}(s,\varphi U\varphi))$. *PTLK* 的语义通过下面定义的满足性关系 $[s,\varphi]_M$ 给出.

定义 4 (*PTLK* 语义)设 M 是定义在 Ap 和 Ag 上的概率 *Kripke* 结构,φ 是 *PTLK* 公式,

$s\in S$ 是 M 中的状态.φ 在 M 中 s 处的满足性关系 $[s,\varphi]_M$ 递归定义如下:

- 对于原子命题 $a\in Ap$,$[s,a]_M:=True$ 当且仅

当 $a\in L(s)$;

- $[s,\neg\varphi]_M=\overline{[s,\varphi]_M}$;
- $[s,\varphi\wedge\varphi]_M=[s,\varphi]_M\wedge[s,\varphi]_M$;
- $[s,\varphi\vee\varphi]_M=[s,\varphi]_M\vee[s,\varphi]_M$;
- $[s,X^{\triangleright p}\varphi]_M=True$ 当且仅当 $\sum_{[s',\varphi]_M=True}P(s,s')\triangleright p$,这里$\triangleright\in\{<,\leqslant,>,\geqslant\}$;
- $[s,\varphi U^{\triangleright p}\varphi]_M=True$ 当且仅当 $Prob(s,\varphi U\varphi)\triangleright p$,这里$\triangleright\in\{<,\leqslant,>,\geqslant\}$;
- $[s,K_i^{\triangleright p}\varphi]_M=True$ 当且仅当 $\sum_{[s',\varphi]_M=True}P_i(s,s')\triangleright p$,这里$\triangleright\in\{<,\leqslant,>,\geqslant\}$.

PTLK 是在时态认知逻辑 *CTLK*[15]的基础上加入概率推理得到的,因此 *CTLK* 可以看成 *PTLK* 的子集.

3 三值抽象

3.1 抽象模型

本文的主要目的是提供一种保持 *PTLK* 真值的抽象技术,其主要原理是合并等价的精确状态,主要特点是抽象系统中状态之间和认知关系上的概率分布利用区间进行表示.我们首先定义抽象模型,然后展示如何从状态空间的等价划分演绎抽象模型.

定义 5 (抽象概率 Kripke 结构) *Ap* 和 *Ag* 上的抽象概率 Kripke 结构 A 是一个多元组 $A=(\hat{S},\hat{s}_0,\widehat{P^l},\widehat{P^u},\widehat{P_1^l},\widehat{P_1^u},\ldots,\widehat{P_n^l},\widehat{P_n^u},\hat{L},?)$,这里

- $\hat{S}$是有限的抽象状态集;
- $\hat{s}_0\in\hat{S}$是初始状态;
- $\widehat{P^l},\widehat{P^u}:\hat{S}\times\hat{S}\to[0,1]$是状态之间转换概率区间的下界和 上界,且满足对任意的$\hat{s},\hat{s}'\in\hat{S}$,$\widehat{P^l}(\hat{s},\hat{S})\leqslant 1,\widehat{P^u}(\hat{s},\hat{S})\leqslant 1,\widehat{P^l}(\hat{s},\hat{s}')\leqslant\widehat{P^u}(\hat{s},\hat{s}'),\widehat{P^l}(\hat{s},\hat{S})\leqslant\widehat{P^u}(\hat{s},\hat{S})$,这里$\widehat{P^l}(\hat{s},\hat{S})=\sum_{\hat{s}'\in\hat{S}}\widehat{P^l}(\hat{s},\hat{s}'),\widehat{P^u}(\hat{s},\hat{S})=\sum_{\hat{s}'\in\hat{S}}\widehat{P^u}(\hat{s},\hat{s}')$;
- $\widehat{P_i^l},\widehat{P_i^u}:\hat{S}\times\hat{S}\to[0,1]$是认知关系 $\sim_i$ 上转换概率区间的下界和上界,且满足对任意的$\hat{s},\hat{s}'\in\hat{S}$,$\widehat{P_i^l}(\hat{s},\hat{S})\leqslant 1,\widehat{P_i^u}(\hat{s},\hat{S})\leqslant 1,\widehat{P_i^l}(\hat{s},\hat{s}')\leqslant\widehat{P_i^u}(\hat{s},\hat{s}'),\widehat{P_i^l}(\hat{s},\hat{S})\leqslant\widehat{P_i^u}(\hat{s},\hat{S})$;
- $\hat{L}:\hat{S}\to 2^{Ap}$是状态标记函数,为每个状态指定该状态下值为真的命题;
- $?:\hat{S}\to 2^{Ap}$标记每个状态下值不确定的原子命题.

令 $H=(P,P_1,\ldots,P_n)$为一概率分布函数的多元组,其中 P 是状态之间的概率分布函数,

$P_i(1\leqslant i\leqslant n)$是认知关系上的概率分布函数.称 $H=(P,P_1,\ldots,P_n)$和抽象概率 Kripke 结构 $A=(\hat{S},\hat{s}_0,\widehat{P^l},\widehat{P^u},\widehat{P_1^l},\widehat{P_1^u},\ldots,\widehat{P_n^l},\widehat{P_n^u},\hat{L},?)$相容当且仅当$\forall\hat{s},\hat{s}'\in\hat{S},\widehat{P^l}(\hat{s},\hat{s}')\leqslant P(\hat{s},\hat{s}')\leqslant\widehat{P^u}(\hat{s},\hat{s}')$,$\forall\hat{s},\hat{s}'\in\hat{S}\,\forall 1\leqslant i\leqslant n,\widehat{P_i^l}(\hat{s},\hat{s}')\leqslant P_i(\hat{s},\hat{s}')\leqslant\widehat{P_i^u}(\hat{s},\hat{s}')$.

定义 6 (抽象模型上的 *PTLK* 语义)令 $H=(P,P_1,\ldots,P_n)$为一概率分布函数的多元组,$A=(\hat{S},\hat{s}_0,\widehat{P^l},\widehat{P^u},\widehat{P_1^l},\widehat{P_1^u},\ldots,\widehat{P_n^l},\widehat{P_n^u},\hat{L},?)$为抽象概率 *Kripke* 结构,$H$ 与 A 相容,φ 是 *PTLK* 公式,$\hat{s}\in\hat{S}$是 A 中的状态.H 上的满足性关系$[\hat{s},a]_{A,H}$递归定义如下:

- 如果 $a\in L(\hat{s})$,则$[\hat{s},a]_{A,H}=T$;如果 $a\in?(\hat{s})$,则$[\hat{s},a]_{A,H}=?$;否则$[\hat{s},a]_{A,H}=\bot$;
- $[\hat{s},\varphi\wedge\varphi]_{A,H}=[\hat{s},\varphi]_{A,H}\wedge[\hat{s},\varphi]_{A,H}$,$[\hat{s},\varphi\vee\varphi]_{A,H}=[\hat{s},\varphi]_{A,H}\vee[\hat{s},\varphi]_{A,H}$,$[\hat{s},\neg\varphi]_{A,H}=\overline{[\hat{s},\varphi]_{A,H}}$;

(这里需要注意 $T\wedge?=?,?\wedge?=?,\bot\wedge?=\bot,T\vee?=T,?\vee?=?,\bot\vee?=?,\overline{T}=\bot,\overline{\bot}=T,\overline{?}=?$)

$?\cdot\quad[\hat{s},X^{\geqslant p}\varphi]_{A,H}=$

$$\begin{cases}T & \text{如果}\sum_{[\hat{s}',\varphi]_{A,H}=T}P(\hat{s},\hat{s}')\geqslant p\\ \bot & \text{如果}\sum_{[\hat{s}',\varphi]_{A,H}=\bot}P(\hat{s},\hat{s}')\geqslant 1-p\\ ? & \text{其它}\end{cases}$$

$[\hat{s},X^{>p}\varphi]_{A,H}=$

$$\begin{cases}T & \text{如果}\sum_{[\hat{s}',\varphi]_{A,H}=T}P(\hat{s},\hat{s}')>p\\ \bot & \text{如果}\sum_{[\hat{s}',\varphi]_{A,H}=\bot}P(\hat{s},\hat{s}')>1-p\\ ? & \text{其它}\end{cases}$$

$[\hat{s},X^{\leqslant p}\varphi]_{A,H}=$

$$\begin{cases}T & \text{如果}\sum_{[\hat{s}',\varphi]_{A,H}=\bot}P(\hat{s},\hat{s}')>1-p\\ \bot & \text{如果}\sum_{[\hat{s}',\varphi]_{A,H}=T}P(\hat{s},\hat{s}')>p\\ ? & \text{其它}\end{cases}$$

$[\hat{s},X^{<p}\varphi]_{A,H}=$

$$\begin{cases} T & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=\perp} P(\hat{s},\hat{s'}) \geqslant 1-p \\ \perp & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=T} P(\hat{s},\hat{s'}) \geqslant p \\ ? & \text{其它} \end{cases};$$

·

$$[\hat{s},\varphi U^{\geqslant p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果 } Prob(\hat{s},\varphi U\varphi,T) \geqslant p \\ \perp & \text{如果 } Prob(\hat{s},\varphi U\varphi,\perp) \geqslant 1-p \\ ? & \text{其它} \end{cases}$$

$$[\hat{s},\varphi U^{>p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果 } Prob(\hat{s},\varphi U\varphi,T) > p \\ \perp & \text{如果 } Prob(\hat{s},\varphi U\varphi,\perp) > 1-p \\ ? & \text{其它} \end{cases}$$

$$[\hat{s},\varphi U^{\leqslant p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果 } Prob(\hat{s},\varphi U\varphi,\perp) > 1-p \\ \perp & \text{如果 } Prob(\hat{s},\varphi U\varphi,T) > p \\ ? & \text{其它} \end{cases}$$

$$[\hat{s},\varphi U^{<p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果 } Prob(\hat{s},\varphi U\varphi,\perp) \geqslant 1-p \\ \perp & \text{如果 } Prob(\hat{s},\varphi U\varphi,T) \geqslant p \\ ? & \text{其它} \end{cases}$$

这里 $Prob(\hat{s},\varphi U\varphi,T) = Prob(Path_A^{\omega}(\hat{s},\varphi U\varphi))$, $Prob(\hat{s},\varphi U\varphi,\perp) = prob(\{\pi \in Path_A^{\omega}(\hat{s}) \mid \pi \text{ 不满足 } \varphi U\varphi\})$;

·

$$[\hat{s},K_i^{\geqslant p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=T} P_i(\hat{s},\hat{s'}) \geqslant p \\ \perp & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=\perp} P_i(\hat{s},\hat{s'}) \geqslant 1-p \\ ? & \text{其它} \end{cases}$$

$$[\hat{s},K_i^{>p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=T} P_i(\hat{s},\hat{s'}) > p \\ \perp & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=\perp} P_i(\hat{s},\hat{s'}) > 1-p \\ ? & \text{其它} \end{cases}$$

$$[\hat{s},K_i^{\leqslant p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=\perp} P_i(\hat{s},\hat{s'}) > 1-p \\ \perp & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=T} P_i(\hat{s},\hat{s'}) > p \\ ? & \text{其它} \end{cases}$$

$$[\hat{s},K_i^{<p}\varphi]_{A,H} =$$

$$\begin{cases} T & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=T} P_i(\hat{s},\hat{s'}) \geqslant 1-p \\ \perp & \text{如果} \sum_{[\hat{s},\varphi]_{A,H}=T} P_i(\hat{s},\hat{s'}) > p \\ ? & \text{其它} \end{cases}.$$

3.2 基于状态空间等价划分的抽象模型的计算

抽象的基本原理是将等价的状态进行合并,下面我们说明如何从状态空间上的等价划分出发,演绎抽象模型.设 $M=(S,s_0,\sim_1,\ldots,\sim_n,P,P_1,\ldots,P_n,L)$ 为概率 Kripke 结构,φ 是 *PTLK* 公式,$\Gamma=\{S_1,\ldots,S_m\}\subseteq 2^S$ 是 S 的一个划分,即 $\forall 1\leqslant i,j\leqslant m, S_i\neq\phi, S_i\cap S_j=\phi(i\neq j), \bigcup_{i=1}^{m} S_i = S$.从 M,φ 和 Γ 演绎出的抽象概率 *Kripke* 结构定义为 $A=(\hat{S},\hat{s_0},\hat{P^l},\hat{P^u},\hat{P_1^l},\hat{P_1^u},\ldots,\hat{P_n^l},\hat{P_n^u},\hat{L},?)$,这里

- $\hat{S}=\{S_1,\ldots,S_m\}$;
- 如果 $s_0\in S_i$,则 $\hat{s_0}=S_i$;
- 对任意的 $S_i,S_j,\hat{P^l}(S_i,S_j)=\min_{s\in S_i}P(s,S_j),\hat{P^u}(S_i,S_j)=\max_{s\in S_i}P(s,S_j)$;
- 对任意的 $1\leqslant k\leqslant n,\hat{P_k^l}(S_i,S_j)=\min_{s\in S_i}P_k(s,S_j),\hat{P_k^u}(S_i,S_j)=\max_{s\in S_i}P_k(s,S_j)$;
- ·

对于任意的 $a\in Ap$,如果 $a\in\bigcap_{s\in S_i}L(s)$,则 $a\in\hat{L}(S_i)$;如果 S_i 中存在不相同的状态 s,s' 使得 $a\in L(s)$,$a\notin L(s')$,则 $a\in?(S_i)$.

3.3 属性保持

抽象的目的是在保持属性的同时能够简化模型,本小节我们将探讨 PTLK 公式的满足性在抽象框架下的保持关系.

定理 1　令 $M=(S,s_0,\sim_1,\ldots,\sim_n,P,P_1,\ldots,P_n,L)$ 为概率 *Kripke* 结构,$\Gamma=\{S_1,\ldots,S_m\}\subseteq 2^S$ 是 S 的一个划分,φ 是 *PTLK* 公式,$A=(\hat{S},\hat{s_0},\hat{P^l},\hat{P^u},\hat{P_1^l},\hat{P_1^u},\ldots,\hat{P_n^l},\hat{P_n^u},\hat{L},?)$是从 Γ 和 φ 演绎出的 M 的抽象模

型，$H=(\widehat{P^l},\widehat{P_1^l},\widehat{P_2^l},\ldots,\widehat{P_n^l})$. 我们有下面两个结论：

1) 如果$[\widehat{s_0},\varphi]_{A,H}=T$，则$[s_0,\varphi]_M=\text{True}$；

2) 如果$[\widehat{s_0},\varphi]_{A,H}=\perp$，则$[s_0,\varphi]_M=\text{False}$.

证明：通过对 φ 的结构进行归纳来完成证明. 对于原子命题，$\neg$ 算子，$\wedge$ 算子和 $\vee$ 算子，结论显然成立，我们主要考察 $X^{\geqslant p}$，$U^{\geqslant p}$，$K_i^{\geqslant p}$ 三个算子（其它算子如 $X^{>p}$，$X^{\leqslant p}$ 等，证明过程类似，故不再考虑）. 首先利用归纳法证明第一个结论.

1)

Case1. $\varphi=X^{\geqslant p}\varphi$

依据定义 6 中定义的$[\widehat{s_0},\varphi]_{A,H}=T$ 的语义，我们有 $\sum_{[\widehat{s'},\varphi]_{A,H}=T}\widehat{P^l}(\widehat{s_0},\widehat{s'})\geqslant p$. 又由 $\widehat{P^l}$ 的定义，$p\leqslant\sum_{[\widehat{s'},\varphi]_{A,H}=T}\widehat{P^l}(\widehat{s_0},\widehat{s'})=\sum_{[\widehat{s'},\varphi]_{A,H}=T}\min_{s\in\widehat{s_0}}P(s,\widehat{s'})\leqslant\sum_{[\widehat{s'},\varphi]_{A,H}=T}P(s_0,\widehat{s'})=\sum_{[\widehat{s'},\varphi]_{A,H}=T}\sum_{s\in\widehat{s'}}P(s_0,s)$. 由归纳假设，$[\widehat{s'},\varphi]_{A,H}=T$ 意味着 $\forall s\in\widehat{s'}$，$[s,\varphi]_M=\text{True}$. 因此 $\sum_{[\widehat{s'},\varphi]_{A,H}=T}\sum_{s\in\widehat{s'}}P(s_0,s)\leqslant\sum_{[s_1,\varphi]_M=True}P(s_0,s_1)$，即 $\sum_{[s_1,\varphi]_M=True}P(s_0,s_1)\geqslant p$.

Case2. $\varphi=\varphi U^{\geqslant p}\gamma$

依据定义 6 中定义的$[\widehat{s_0},\varphi U^{\geqslant p}\gamma]_{A,H}=T$ 的语义，我们有 $Prob(\hat{s},\varphi U\gamma,T)\geqslant p$，即 $Prob(path_M^{\omega}(\widehat{s_0},\varphi U\gamma))=\sum_{\widehat{s_0}\ldots|=\varphi}\sum_{i\geqslant 0}\widehat{P^l}(\widehat{s_i},\widehat{s_{i+1}})\geqslant P$，这里$\widehat{s_0}\ldots|=\varphi U\gamma$ 表示从 $\widehat{s_0}$ 出发满足 $\varphi U\gamma$ 的路径.

由 $\widehat{P^l}$ 的定义，$\sum_{\widehat{s_0}\ldots|=\varphi}\sum_{i\geqslant 0}\widehat{P^l}(\widehat{s_i},\widehat{s_{i+1}})=\sum_{\widehat{s_0}\ldots|=\varphi}\sum_{i\geqslant 0}\min_{s_i\in\widehat{s_i}}P(s_i,\widehat{s_{i+1}})=\sum_{\widehat{s_0}\ldots|=\varphi}\sum_{i\geqslant 0}\min_{s_i\in\widehat{s_i}}\sum_{s_{i+1}\in\widehat{s_{i+1}}}P(s_i,s_{i+1})$. 设路径$\widehat{s_0}\,\widehat{s_1}\ldots\widehat{s_n}\ldots$ 满足 $\varphi U\gamma$，且$[\widehat{s_n},\gamma]_{A,H}=True$，$[\widehat{s_i},\varphi]_{A,H}=True(0\leqslant i<n)$. 由归纳假设可知 $\forall s_n\in\widehat{s_n}[s_n,\gamma]_M=True$，$\forall s_i\in\widehat{s_i}[s_i,\varphi]_M=True(0\leqslant i<n)$，即路径 $s_0s_1\ldots s_n\ldots$ 满足 $\varphi U\gamma$. 因此 $p\leqslant\sum_{\widehat{s_0}\ldots|=\varphi}\sum_{i\geqslant 0}\min_{s_i\in\widehat{s_i}}\sum_{s_{i+1}\in\widehat{s_{i+1}}}P(s_i,s_{i+1})\leqslant\sum_{s_0\ldots|=\varphi}\sum P(s_i,s_{i+1})=Prob(s_0,\varphi U\gamma)$，即$[s_0,\varphi U\gamma]_M=True$.

Case3. $\varphi=K_i^{\geqslant p}\varphi$

依据定义 6 中定义的$[\widehat{s_0},\varphi]_{A,H}=T$ 的语义，我们有 $\sum_{[\widehat{s'},\varphi]_{A,H}=T}\widehat{P_i^l}(\widehat{s_0},\widehat{s'})\geqslant p$. 又由 $\widehat{P_i^l}$ 的定义，$p\leqslant\sum_{[\widehat{s'},\varphi]_{A,H}=T}\widehat{P_i^l}(\widehat{s_0},\widehat{s'})=\sum_{[\widehat{s'},\varphi]_{A,H}=T}\min_{s\in\widehat{s_0}}P_i(s,\widehat{s'})\leqslant\sum_{[\widehat{s'},\varphi]_{A,H}=T}P_i(s_0,\widehat{s'})=\sum_{[\widehat{s'},\varphi]_{A,H}=T}\sum_{s\in\widehat{s'}}P_i(s_0,s)$.

由归纳假设$[\widehat{s'},\varphi]_{A,H}=T$ 意味着 $\forall s\in\widehat{s'}$，$[s,\varphi]_M=True$. 因此 $p\leqslant\sum_{[\widehat{s'},\varphi]_{A,H}=T}\sum_{s\in\widehat{s'}}P_i(s_0,s)\leqslant\sum_{[s_1,\varphi]_M=True}P_i(s_0,s_1)$，即$[s_0,K_i^{\geqslant p}\varphi]_M=True$.

2)对于第二个结论，归纳证明过程和第一个类似，因此这里不再赘述.

4 概率时态认知逻辑模型检测算法

本小节我们讨论 *PTLK* 的模型检测算法问题，即给定一个抽象概率 Kripke 结构 A 和 *PTLK* 公式 φ，判断 φ 在$\widehat{s_0}$处是否成立. 算法的基本思想基于计算树时态逻辑 *CTL* 的模型检测算法：采用标记算法为每个状态标记在此状态下为真的公式的集合 $label_T(\hat{s})$，值不确定的公式集 $label_?(\hat{s})$，以及值为假的公式集 $label_\perp(\hat{s})$. 初始的时候 $a\in label_T(\hat{s})$ 当且仅当 $a\in L(\hat{s})$，$a\in label_?(\hat{s})$ 当且仅当 $a\in?(\hat{s})$，$a\in label_\perp(\hat{s})$ 当且仅当 $a\notin L(\hat{s})\cup?(\hat{s})$. 在第 i 阶段，具有 $i-1$ 层嵌套算子的子公式将被处理，处理过的子公式将被增加到相应的状态标记下. 一旦算法终止，我们有下面的结论$[\hat{s},\varphi]_{A,H}=True$ 当且仅当$\varphi\in label_T(\hat{s})$，$[\hat{s},\varphi]_{A,H}=?$ 当且仅当 $\varphi\in label_?(\hat{s})$，$[\hat{s},\varphi]_{A,H}=\perp$ 当且仅当 $\varphi\in label_\perp(\hat{s})$.

对于原子命题，否定算子 $\neg$，合取算子 $\wedge$，析取算子 $\vee$，以及算子 $X^{\triangleright p}\varphi$ 与 $\varphi U^{\triangleright p}\varphi$，检测过程可参考文献[12]，这里不再赘述. 对于算子 $K_i^{\geqslant p}\varphi$，由其语义只需检测当前状态与认知关系中满足 φ 的状态之间的概率，具体过程如 Procedure1 所示（以 $K_i^{\geqslant p}\varphi$ 为例）.

Procedure1 $Check(K_i^{\geqslant p}\varphi)$

While $\hat{S}\neq\phi$ do

Choose $\hat{s}\in\hat{S}$

If $\sum_{\varphi\in label_T(\hat{s})}P_i(\hat{s},\widehat{s'})\geqslant p$ then $label_T(\hat{s}):=label_T(\hat{s})\cup\{K_i^{\geqslant p}\varphi\}$

Else

If $\sum_{\varphi\in label_\perp(\hat{s})}P_i(\hat{s},\widehat{s'})\geqslant 1-p$ then $label_\perp(\hat{s}):=$

$label_{\perp}(\hat{s}) \cup \{K_i^{\geqslant p}\varphi\}$

Else $label_{?}(\hat{s}) := label_{?}(\hat{s}) \cup \{K_i^{\geqslant p}\varphi\}$

End if

End if

$\hat{S} := \hat{S} \setminus \{\hat{s}\}$

End While

5 抽象模型的求精

在抽象模型中如果$[\hat{s_0}, \varphi]_{A,H} = ?$,则我们无法确定原始模型是否满足 φ,此时我们需要对抽象模型进行求精.本节我们首先分析抽象失败的原因,然后依据失败的原因给出利用证据和反例引导的求精方法.

5.1 抽象失败原因分析

考察图 1 中的 M 是否满足属性$\varphi = X^{\geqslant \frac{2}{3}} q$. A_1 是 M 的初始抽象模型,由检测算法可知

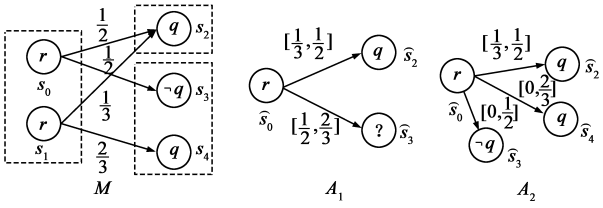

图1　概率Kripke结构的抽象和求精

对任何与 A_1 相容的概率分布 H, $\frac{1}{3} \leqslant \sum_{[\hat{s},q]_{A,H}=T} P(\hat{s},\hat{s'}) \leqslant \frac{1}{2}$, $\sum_{[\hat{s},q]_{A,H}=\perp} P(\hat{s},\hat{s'}) = 0$,因此$[\hat{s_0}, X^{\geqslant \frac{2}{3}} q]_{A,H} = ?$.造成$[\hat{s_0}, X^{\geqslant \frac{2}{3}} q]_{A,H} = ?$的原因有两个:

1)状态标记函数不精确:A_1 中的抽象状态$\hat{s_3}$包含 s_3 和 s_4,而 $q \in L(s_4)$, $q \notin L(s_3)$,因此 $q \in ?(\hat{s_3})$.在这种状态空间划分下,对于初始状态 s_0 而言,属性 Xq 不成立的概率从$\frac{1}{2}$变为不可知,从而导致 $\sum_{[\hat{s},q]_{A,H}=\perp} P(\hat{s},\hat{s'}) = 0$.

2)概率转换不精确:考察图 1 中的抽象模型 A_2(由将 A_1 中的$\hat{s_3}$划分成 s_3 和 s_4 所得). A_2 中初始状态$\hat{s_0}$和$\hat{s_2}$之间的转换概率为区间$[\frac{1}{3}, \frac{1}{2}]$,而实际上原始模型中初始状态 s_0 和 s_2 之间的转换概率为$\frac{1}{2}$.取概率分布 $H = \{P, P_1\}$,这里 $P(\hat{s_0}, \hat{s_2}) = \frac{1}{3}$, $P(\hat{s_0}, \hat{s_4}) = 0$, $P(\hat{s_0}, \hat{s_3}) = 0$.此时$[\hat{s_0}, X^{\geqslant \frac{2}{3}} q]_{A,H} = ?$,因此需要进一步对抽象状态$\hat{s_0}$进行求精.

对于第一种情况可以通过对抽象状态进行进一步划分来避免,如在图 1 的 A_1 中将$\hat{s_3} = \{s_3, s_4\}$分解成$\{s_3\}$和$\{s_4\}$(如图 1 的 A_2 所示).对于第二种情况,原则上也可以通过对抽象状态进行进一步划分来避免,但是如何选择抽象状态进行进一步划分对提高验证的效率是至关重要的.在下一小节我们将讨论如何利用验证的结果来引导求精.

5.2 抽象求精

本小节我们将探讨如何利用模型检测的结果来引导抽象系统的求精.

定义 7 (最小证据)给定公式 φ,称有限路径 π 是满足 φ 的最小路径当且仅当 π 的任一前缀(π 除外)都不满足 φ.

定义 8 (最小反例)给定公式 φ,称有限路径 π 是不满足φ 的最小反例当且仅当π 是满足 $\neg\varphi$ 的最小证据.

满足 φ 的最小证据的集合称为 φ 的最小证据集,满足 $\neg\varphi$ 的最小证据的集合称为φ 的最小反例集.

定义 9 (语法树) *PTLK* 公式φ 的语法树是一棵树,其中内部结点标记为算子 $\neg, \wedge, \vee, X^{\triangleright p}, U^{\triangleright p}, K_i^{\triangleright p}$,终端结点标记为原子命题,这里$\triangleright \in \{<, \leqslant, >, \geqslant\}$.

5.2.1 计算最小证据集

假设模型检测算法已经运行完毕,即每个状态均已经标记上了该状态下值为真,为假以及不确定的公式集.在此基础上计算最小证据集的方法 *compute-witness*$_{\min}$如算法 1 所示.

算法 1 *compute-witness*$_{\min}(v, \hat{s})$

If $op(v) = X^{\triangleright p}$ then return *compute-witness*$_{\min} X^{\triangleright p}(v, \hat{s})$;

If $op(v) = U^{\triangleright p}$ then return *compute-witness*$_{\min} U^{\triangleright p}(v, \hat{s})$;

If $op(v) = K_i^{\triangleright p}$ then return *compute-witness*$_{\min} K_i^{\triangleright p}(v, \hat{s})$;

If $op(v) = \wedge$ then return *compute-witness*$_{\min}(v.left, s) \otimes$ *compute-witness*$_{\min}(v.right, s)$;

If $op(v) = \vee$ then return *compute-witness*$_{\min}(v.left, s)$ 8 *compute-witness*$_{\min}(v.right, s)$.

对于否定算子 $\neg$,由 *PTLK* 的语义定义,在不失表达力的情况下可假设 $\neg$ 仅仅作用于原子命题,例如 $\neg K_i^{\geqslant p}\varphi \equiv K_i^{<p} \neg \varphi$.因此在算法 1 中仅考虑了其余的五个算子,具体的最小证据集计算过程需分五种情况讨论,其中关于算子 $X^{\triangleright p}, U^{\triangleright p}, \wedge, \vee$ 可参考文献[12].现在考察 $K_i^{\triangleright p}$算子.设当前状态为$\hat{s}$,依据 K_i 算子的语义对所

有满足 φ 的状态$\widehat{s'}$，如果 $P_i(\hat{s},\widehat{s'})>0$，则$\hat{s}\widehat{s'}$是满足 $K_i\varphi$ 的最小证据.

compute-witness$_{\min} K_i^{\triangleright p}(v,\hat{s})$

While $sat(\varphi)\neq\phi$

Choose $\widehat{s'}\in sat(\varphi)$

If $P_i(\hat{s},\widehat{s'})>0$ then print$(\hat{s},\widehat{s'})$ Endif

$sat(\varphi)=sat(\varphi)\setminus\{\widehat{s'}\}$

End while

5.2.2 计算最小反例集

计算 φ 的最小反例集，等价于计算 $\neg\varphi$ 的最小证据集，因此 *compute-counterexample*$_{\min}(\varphi)$ = *compute-witness*$_{\min}(\neg\varphi)$.

5.2.3 最小证据和反例引导的求精

利用算法 1 可以得到最小证据和反例集，现在来探讨如何利用这些证据和反例引导抽象求精. 设 $\pi=\hat{s}_0\ldots\hat{s}_n$是最小证据或者反例，求精过程为：先对$\widehat{s_{n-1}}$进行求精，得到新的抽象状态$\widehat{s^1_{n-1}}$和$\widehat{s^2_{n-1}}$，此时如果属性成立则返回，否则继续对$\widehat{s^1_{n-1}}$和$\widehat{s^2_{n-1}}$求精，当对$\widehat{s_{n-1}}$无法求精后，进一步对$\widehat{s_{n-2}}$求精. 当对整个路径 π 求精失败后，重新选择一条最小证据或者反例进行求精. 求精过程终止于属性成立或者抽象状态和原始状态一致. 对$\widehat{s_{n-1}}$的求精原则为：计算$\widehat{s_{n-1}}$包含的状态集中到$\hat{s}_n$转换概率为区间下界的那些状态，然后将这些状态分离出去形成新的状态$\widehat{s^1_{n-1}}$，$\widehat{s_{n-1}}$中剩余的状态形成新的状态$\widehat{s^2_{n-1}}$. 对$\widehat{s_{n-1}}$具体求精的算法如下：

算法 2 *refinement*$(\pi=\hat{s}_0\ldots\hat{s}_n)$

$W_1=W_2=\phi$

i 是满足所包含状态数目不低于 2 的抽象状态的最大的下标

While $\hat{s}_i\neq\phi$

Choose $s\in\hat{s}_i$

If $P(s,\hat{s}_n)=l$ then $W_1=W_1\cup\{s\}$//l 为$\widehat{s_{n-1}}$到$\hat{s}_n$转换概率区间的下界

Else $W_2=W_2\cup\{s\}$

End if

$\hat{s}_i=\hat{s}_i\setminus\{s\}$

End while

Return $\pi_1=\hat{s}_0\ldots\widehat{s_{i-1}}W_1\widehat{s_{i+1}}\ldots\hat{s}_n$，$\pi_2=\hat{s}_0\ldots\widehat{s_{i-1}}W_2\widehat{s_{i+1}}\ldots\hat{s}_n$

在此求精下得到两条路径 $\pi_1=\hat{s}_0\ldots\widehat{s_{i-1}}W_1\widehat{s_{i+1}}\ldots\hat{s}_n$，$\pi_2=\hat{s}_0\ldots\widehat{s_{i-1}}W_2\widehat{s_{i+1}}\ldots\hat{s}_n$. 如果此时抽象系统满足属性，则求精结束，否则继续对 π_1，π_2 求精. 整体的求精思想是对每一条路径逐步求精，直到完成验证过程. 整个求精算法如下：

算法 3 *refinement*(A,φ)

$W=$ *comput-witness*$_{\min}(v,\hat{s}_0)\cup$ *compute-counterexample*$_{\min}(v,\hat{s}_0)$

While $W\neq\phi$

Choose $\pi\in W$

If *refinement*$(\pi)=\{\pi_1,\pi_2\}$ then $W=W\cup\{\pi_1,\pi_2\}$

Else goto L

End if

If the truth of φ in A_1 can be decided then return the truth of φ End if L$W=W\setminus\{\pi\}$

End while

6 模型检测 Dining Cryptographer 协议

6.1 Dining Cryptographer 协议

Dining Cryptographer 协议[13,14]是为匿名广播而设置的协议，David Chaum 是通过下面故事来引入 Dining Cryptographer 协议的：

三个保密家到他们平时喜欢的三星级酒店里吃饭，服务员告诉他们付款规则是采用匿名方式进行的，其中一个保密家可以为这顿饭付账或者由 NSA(美国国家安全局)替他们付账. 三个保密家珍重每个人的匿名付账权力，但是他们疑惑是否由 NSA 来替他们付账.

为了搞清楚谁付账，David Chaum 提出下面的协议规则，该协议预先假设至多只有一个保密家为这顿饭付账：

(*a*)每个保密家将一枚硬币向上抛，硬币落在他和位于其右边的保密家之间，从而只有他和他右边的可以看到硬币的表面.

(*b*)每个保密家然后大声说出他所见到的二个硬币，一个是自己抛的，另一个是他左边的人抛的，说出他看到的二个硬币表面图案一样或不一样.

(*c*)如果其中的某一个保密家支付了款项，他就说出相反的结果，如果他没有支付这顿饭钱他应该照实说出.

依据这个协议，所有的保密家可以根据他们所说出的情况，判断出是由 *NSA* 付账还是他们中的某一个人付账：当不一样数为偶数时是由 *NSA* 付账，不一样数为奇数时由他们中的某一个人付账. 最为重要的是当

由他们中的某一个人付账时,不知到具体是哪一个.

6.2 Dining Cryptographer 协议的概率 Kripke 结构

每一个保密家的状态涉及到的变量主要包括:谁付的帐,硬币表面图案,保密家看到的二个硬币表面图案一样还是不一样.以保密家 1 为例,相应的变量定义如下:

· $coin_1 \in \{0,1,2\}$:所抛硬币哪面朝上,0 表示还没有抛硬币;

· $agree_1 \in \{0,1,2\}$:保密家看到的二个硬币表面图案一样或不一样,1 表示不一样,2 表示一样,0 表示硬币还没抛;

对于保密家 2 和保密家 3,相应的变量为 $coin_i$, $agree_i$($i \in \{2,3\}$).另外,$pay \in \{0,1,2,3\}$表示谁付的帐,其中 $pay=0$ 表示由 *NSA* 付账.每个保密家的状态转换以"条件→结果:p"的形式给出,这里 p 表示转换的概率.保密家 1 执行的动作序列可以概括为:抛硬币,依据看到的二个硬币表面图案说出结果.依据该动作序列,我们得到下面的状态转换关系:

初始状态:$coin_1=0 \wedge agree_1=0 \wedge pay=1$(保密家 1 付账)或者 $coin_1=0 \wedge agree_1=0 \wedge pay \neq 1$(保密家 1 没有付账).

状态转换:$coin_1=0 \rightarrow coin_1=1:0.5$;

$coin_1=0 \rightarrow coin_1=2:0.5$;

$coin_1>0 \wedge coin_2>0 \wedge coin_1=coin_2 \wedge pay=1 \rightarrow agree_1=1:1$;

$coin_1>0 \wedge coin_2>0 \wedge coin_1=coin_2 \wedge pay \neq 1 \rightarrow agree_1=2:1$;

$coin_1>0 \wedge coin_2>0 \wedge coin_1 \neq coin_2 \wedge pay=1 \rightarrow agree_1=2:1$;

$coin_1>0 \wedge coin_2>0 \wedge coin_1 \neq coin_2 \wedge pay \neq 1 \rightarrow agree_1=1:1$.

认知关系 $\sim_1$:对于全局状态 s, s',如果 s, s'下 $coin_1=coin'_1$, $agree_1=agree'_1$,且 pay, pay'满足 $pay=pay'=1$ 或者 $pay \neq 1$, $pay' \neq 1$,则认为两者满足认知关系,且如果有 n 个状态(包括 s)和 s 满足认知关系 $\sim_1$,则 $P_1(s,s')=\frac{1}{n}$.

对于保密家 2 和保密家 3,状态转换关系和保密家 1 类似,只需对上述关系进行适当替换即可.现在考虑这样的安全需求:如果保密家 1 没有付账的话,则保密家 1 知道下述事实的概率不低于$\frac{2}{3}$:要么其他保密家也没有付账,要么保密家 2 和保密家 3 当中肯定有一个付账了,但不知道具体是哪一个(事实上保密家 1 知道该事实的概率应该为 1,这里设置概率为$\frac{2}{3}$是为了方便说明抽象技术).该安全需求利用 *PTLK* 表示为 $\varphi = pay \neq 1 \rightarrow K_1^{\geqslant \frac{2}{3}}(pay=0) \vee (K_1^{\geqslant \frac{2}{3}}(pay \neq 0 \rightarrow (pay=2 \vee pay=3)) \wedge \neg K_1^{\geqslant \frac{2}{3}}(pay \neq 0 \rightarrow pay=2) \wedge \neg K_1^{\geqslant \frac{2}{3}}(pay \neq 0 \rightarrow pay=3))$.

6.3 建立抽象模型

原始模型中的精确状态是对变量 $coin_1$, $coin_2$, $coin_3$, $agree_1$, $agree_2$, $agree_3$, pay 的一组赋值.因为 φ 中仅包含变量 pay,所以对状态空间划分的标准可定义为:s 与 s'等价当且仅当在状态 s 和 s'下 pay 的值相等.pay 共有四种不同的取值,因此抽象系统共有四个抽象状态,如图 2 所示.由于 φ 中仅包含认知算子 K_1,因此图 2 中只显示了认知关系 $\sim_1$ 上的概率分布.

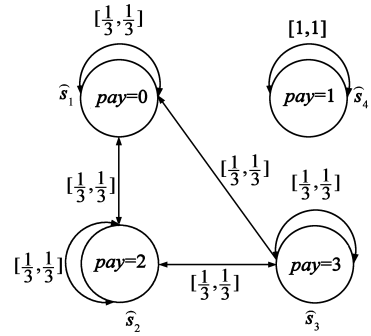

图2　Ding Cryptographer协议抽象模式

现在考察为什么图 2 中认知关系 $\sim_1$ 上的概率分布是$[\frac{1}{3},\frac{1}{3}]$和$[1,1]$.对于保密家 1 和对 $coin_1$, $agree_1$ 的任意赋值 $coin_1=i$, $agree_1=j$ 而言,使得 $coin_1=i$, $agree_1=j$, $pay=0$, $coin_1=i$, $agree_1=j$, $pay=2$, $coin_1=i$, $agree_1=j$, $pay=3$ 的状态 s_1, s_2, s_3 是 $\sim_1$ 等价的.在三个状态下,尽管 pay 的值不相同,但是由于保密家 1 不知道 pay 到底等于几,因此从保密家 1 的角度来看,三个状态是不可区分的,因此 $P_1(s_1,s_1)=P_1(s_2,s_2)=P_1(s_3,s_3)=P_1(s_1,s_2)=P_1(s_2,s_1)=P_1(s_1,s_3)=P_1(s_3,s_1)=P_1(s_2,s_3)=P_1(s_3,s_2)=\frac{1}{3}$.依据状态空间的等价划分原则:在状态 s 和 s'下如果 pay 的值相等,则 s 与 s'等价,s_1, s_2, s_3 属于不同的等价类.依据抽象模型的定义$\widehat{P_1}(\widehat{s_1},\widehat{s_1})=\widehat{P_1}(\widehat{s_2},\widehat{s_2})=\widehat{P_1}(\widehat{s_3},\widehat{s_3})=\widehat{P_1}(\widehat{s_1},\widehat{s_2})=\widehat{P_1}(\widehat{s_2},\widehat{s_1})=\widehat{P_1}(\widehat{s_1},\widehat{s_3})=\widehat{P_1}(\widehat{s_3},\widehat{s_1})=\widehat{P_1}(\widehat{s_2},\widehat{s_3})=\widehat{P_1}(\widehat{s_3},\widehat{s_2})=\frac{1}{3}$而对于使得 $coin_1=i$, $agree_1=j$, $pay=1$ 的状态 s_4, s_5, $\widehat{s_4}=\widehat{s_5}$,因此$\widehat{P_1}(\widehat{s_4},\widehat{s_4})=1$.运用第四部分的模型检测算法,我们发现抽象系统满足 φ,因此原系统也满足 φ.

6.4 实验结果

我们以 Java 语言开发了 Dining Cryptographer 协议检测工具 DCcheck.测试的平台为:2.10GHZ Intel Core2 Duo Cpu,2GB RAM,Windows Vista 操作系统.对于具有 n 个保密家的协议,验证的属性为:

$$\varphi = pay \neq 1 \rightarrow K_1^{\geqslant \frac{2}{3}}(pay = 0) \vee (K_1^{\geqslant \frac{2}{3}}(pay \neq 0 \rightarrow (\bigvee_{i=2}^{n} pay = i)) \wedge \bigwedge_{i=2}^{n} \neg K_1^{\geqslant \frac{2}{3}}(pay \neq 0 \rightarrow pay = i)).$$

表 1 为测试的实验结果,其中 n 表示保密家的数目,S 表示原系统中精确状态的数目,t_1 表示验证原系统满足 φ 所消耗的时间(单位为秒,值取五次实验的平均值),m_1 表示验证原系统满足 φ 所需要的内存空间的顶峰值(单位为 K,值取五次实验的平均值),S_a 表示抽象系统中状态的数目,t_2 表示验证抽象系统满足 φ 所消耗的时间(单位为秒,值取五次实验的平均值),m_2 表示验证抽象系统满足 φ 所需要的内存空间的顶峰值(单位为 K,值取五次实验的平均值).

从实验结果我们可以发现,三值抽象技术可以降低内存需求大约 30%,降低验证时间 25%.因此我们的抽象技术在一定程度上缓解了状态空间爆炸问题.

表 1 三值抽象技术在模型检测 *Dining Cryptographer* 协议中的应用

n	11	12	13	14	15	16
S	49164	106509	229390	491535	1048592	*N/A*
t_1	396.813	1002.935	2342.258	5602.335	14758.684	*N/A*
m_1	20616	23120	30004	53316	81340	*N/A*
S_a	12	13	14	15	16	*N/A*
t_2	302.813	790.466	1890.260	4033.681	10847.633	*N/A*
m_2	14618	16867	22023	36795	63692	*N/A*

对于状态空间大小为 k 的系统来说直接进行模型检测需要遍历 k 个状态.现在假设定义了 h 个等价类,得到的抽象模型只有 h 个状态,在抽象模型上再进行模型检测只需遍历 h 个状态.因为 h 远远小于 k,所以在理论上我们的抽象技术可以大幅降低内存空间的需求,但是在实际验证中仅仅降低了 30% 左右.造成理论和实际差距较大的主要原因在于在计算抽象状态时,需要搜索并存储部分具体的状态.未来我们将进一步探讨抽象状态的计算.

7 结论

为克服模型检测概率时态认知逻辑 *PTLK* 中的状态空间爆炸问题,文中展示了一种三值抽象技术.我们的抽象技术实现了 *PTLK* 公式可满足和不可满足性的保守计算,即如果抽象系统满足(不满足) *PTLK* 属性,则原始系统也满足(不满足) *PTLK* 属性.且当抽象系统对 *PTLK* 属性的满足性关系不确定时,我们设计了以最小证据和反例引导的求精过程,从而保证了抽象技术的完备性.最后通过模型检测 *Dining Cryptographer* 协议,说明了抽象技术能够有效地降低验证系统的规模,减少验证时间.将来的工作主要包括:1)我们提出的求精过程针对的是具体的路径,效率比较低,未来拟进一步研究如何以最小证据和反例集为单位引导抽象系统的求精;2)在检测工具 *DCcheck* 中,计算抽象状态时需要搜索部分精确的状态.拟研究新的计算方法,降低对精确状态的需求,从而缩小理论和实际之间的差距.

参考文献

[1] E M Clarke, O Grumberg, D Peled. Model Checking[M]. MIT Press, 1999.

[2] 林惠民,张文辉.模型检测:理论、方法与应用[J].电子学报,2002,30 (12A):1907 – 1912.
Lin Huimin, Zhang Wenhui. Model checking: theories, techniques and applications[J]. Acta Electronica Sinica, 2002, 30 (12A):9 – 14(in Chinese).

[3] N Ferreira, M Fisher, W Hoek. Practical Reasoning for Uncertain Agents[J]. Lecture Notes in Computer Science, 2004, 3229:82 – 94.

[4] Z Cao. Model checking for epistemic and temporal properties of uncertain agents[J]. Lecture Notes in Computer Science, 2006, 4088:46 – 58.

[5] R E Bryant. Graph-based algorithms for boolean function manipulation[J]. IEEE Transaction on Computers, 1986, 35(8): 687 – 691.

[6] P Wolper, P Godefroid. Partial order methods for temporal verification[J]. Lecture Notes in Computer Science, 1993, 715:233 – 246.

[7] E A Emerson, A P Sistla. Symmetry and model checking[J]. Formal Methods in System Design, 1996, 9(1):105 – 131.

[8] E M Clarke, O Grumberg, D E Long. Model checking and abstraction[J]. ACM Transactions on Programming Languages and Systems, 1992, 16 (5):1512 – 1542.

[9] 屈婉霞,李暾,郭阳,杨晓东.谓词抽象技术研究[J].软件学报,2008,19(01):27 – 38.

Qu Wanxia, Li Tun, Guo Yang, Yang Xiaodong. Advances in Predicate Abstraction[J]. Journal of Software, 2008, 19(01): 27 – 38. (in Chinese)

[10] O Grumberg. 3-Valued Abstraction for (Bounded) Model Checking[J]. Lecture Notes in Computer Science, 2009, 5799: 21 – 21.

[11] E M Clarke, O Grumberg, S Jha, Y Lu, H Veith. Counterexample-guided abstraction refinement for symbolic model checking[J]. Journal of the ACM, 2003, 50(3): 752 – 794.

[12] J P Katoen, D Klink, M Leucker, V Wolf. Three-valued abstraction for probabilistic systems[J]. J Log Algebr Program, 2012, 81(4): 356 – 389.

[13] D Chaum. The dining cryptographers problem: unconditional sender and recipient untraceability[J]. J Cryptology, 1988, 1(1): 65 – 75.

[14] R Meyden, K Su. Symbolic model checking the knowledge of the dining cryptographers[A]. In Proc. 17th IEEE Computer Security Foundations Workshop[A]. IEEE Computer Society, 2004. 280 – 291.

[15] 骆翔宇,苏开乐,杨晋吉.有界模型检测同步多智体系统的时态认知逻辑[J].软件学报,2006,17(12):2485 – 2498.
Luo Xiangyu, Su Kaile, Yang Jinji. Bounded model checking for temporal epistemic logic in synchronous multi-agent systems[J]. Journal of Software, 2006, 17(12): 2485 – 2498. (in Chinese)

作者简介

周从华　男,1978 年生,江苏大丰人.2006 年在南京大学获得博士学位,现为江苏大学计算机科学与通信工程学院副教授.主要研究方向为模型检测,访问控制,模态逻辑.

E-mail: chzhou@ujs.edu.cn