软件学报 ISSN 1000-9825,CODEN RUXUEW
Journal of Software,[doi: 10.13328/j.cnki.jos.005462]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

异构多智能体系统模型检查*

张业迪, 宋 富

(上海科技大学信息科学与技术学院,上海 201210)

通讯作者: 宋富,E-mail: songfu@shanghaitech.edu.cn

摘要: 近几年,模型检查作为一种自动化系统验证方法,已被应用于多智能体系统的验证.由此延伸出的规约描述语言——交替时态逻辑(ATL),也被给予了高度关注.根据智能体是否可以看到全局信息分为不完全信息和完全信息;根据智能体是否可以记录历史信息分为无记忆能力和无限记忆能力,提出了四种经典的策略类型.这些策略类型是通过 ATL 的语义进行刻画的.然而,在一个多智能体系统中,考虑完全信息和无限记忆能力时,所有智能体都只能选择这一种策略类型;在考虑不完全信息或无记忆能力时,仅在联合模态操作〈〈A〉〉φ 和[[A]]φ 的 A 里出现的智能体具备这种策略类型,而其他智能体还是完全信息和无限记忆能力策略类型.这可能会导致嵌套联合模态操作中智能体策略类型的不一致,且智能体策略类型取决于逻辑公式和逻辑的语义.而在实际多智能体系统中,智能体的策略类型往往取决于系统本身,且不同智能体具有不同的策略类型,即智能体策略类型是异构的,这种多智能体系统被称为异构多智能体系统.针对这些问题,本文提出了一种在语法层对智能体策略类型进行刻画的系统模型,即带类型解释系统.带类型解释系统在已有的解释系统中为每个智能体引入"策略类型"这一属性,允许不同智能体具备不同的策略类型.带类型解释系统可用于异构多智能体系统的建模.针对新提出的系统模型,对 ATL 语义进行了研究,设计了 ATL 模型检查算法,实现了相应的模型检查工具 ShTMC.

关键词: 模型检查;多智能体系统;交替时态逻辑;策略类型

中图法分类号: TP311

中文引用格式:张业迪,宋富.异构多智能体系统模型检查.软件学报,2018,29(6). http://www.jos.org.cn/1000-9825/5462.htm

英文引用格式:Zhang YD, Song F. Model-Checking for Heterogeneous Multi-agent Systems. Ruan Jian Xue Bao/Journal of Software, 2018,29(6) (in Chinese).http://www.jos.org.cn/1000-9825/5462.htm

Model-Checking for Heterogeneous Multi-agent Systems

ZHANG Ye-Di, SONG Fu

(School of Information Science and Technology, Shanghai Tech University,Shanghai201210,China)

Abstract: Model-checking, an automatic verification methodology, has been applied to verify multi-agent systems. Alternating-time temporal logic (ATL),a property specification language for multi-agent systems was also investigated. According to whether agents are able to observe the whole information of the system and whether agents are able to record the history information, agents' strategies are divided into four types. These strategy abilities are defined in semantic level of ATL, as well as other logics. However, under perfect information and perfect recall setting, all agents have the same strategy ability. Under imperfect information and/or imperfect recall setting, only agents appeared in coalition modalities 〈〈A〉〉φ and [[A]]φ have imperfect information and/or imperfect recall strategies, while other agents not in A still have perfect information and perfect recall strategies. When coalition modalities are nested, same agents may have different strategy abilities to fulfill different subformulas, which results in inconsistent. On the other hand, in practice, agents'

* 基金项目: 国家自然科学基金(61402179, 61532019)
Foundation item: National Natural Science Foundation of China (61402179, 61532019)
本文由形式化方法的理论基础专刊特约编辑傅育熙教授、李国强副教授、田聪教授推荐.
收稿时间: 2017-06-29; 修改时间: 2017-09-01; 采用时间: 2017-11-06; jos 在线出版时间: 2017-12-28

strategy abilities are usually decided by the multi-agent systems rather than the specifications, and different agents may own different strategy abilities. This kind of multi-agent systems is called heterogeneous mutli-agent systems. To overcome these problems, in this paper, we propose a new formal model, called typed interpreted systems which are able to define agent's strategy abilities at syntax level. Typed interpreted systemsextend interpreted systems by associating each agent with a strategy type denoting the agent's strategy ability, therefore are able to model heterogeneous mutli-agent systems. We investigate the semantics of ATL on this new model and propose an EXPTIME model-checking algorithm. The model-checking algorithm is implemented in a prototype tool ShTMC.

Key words: model-checking; multi-agent system; alternating-time temporal logic; strategy abilities

模型检查(model-checking)作为一种自动化系统验证方法,已广泛应用于协议、硬件和软件验证[1].它可以将某个实际验证对象转化为数学模型,利用形式化语言将待验证的问题描述为系统规约,最后通过模型检查算法来证明系统模型是否满足系统规约.在模型检查研究初期,主要考虑封闭式系统(closed system).对于这种系统而言,所有可能的行为包括不确定性都被该系统当前状态决定,外部环境不影响系统的行为.封闭式系统常采用 Kripke 结构建模,采用线性时态逻辑 LTL 和计算树时态逻辑 CTL 等形式化语言描述系统规约,对应的模型检查算法也已经被提出和应用[1,2].

然而,面向封闭式系统的模型检查方法不适用于开放式系统(open system)的验证.与封闭式系统不同,开放式系统的行为不仅依赖于系统的当前状态,同时也依赖于系统与环境的交互,开放式系统中的不确定性选择往往取决于环境.1996 年,Kupferman 和 Vardi 将面向封闭式系统的模型检查方法扩展到了开放式系统[3],该方法称为模块检查(module-checking).在模块检查框架中,开放式系统采用一种类似 Kripke 结构的有限状态系统建模,称为模块(module),其状态包括系统状态和环境状态,分别描述系统本身的状态和环境的状态.一个开放式系统满足一个规约当且仅当无论环境怎么选择,系统的行为都要满足规约.

近几年,模型检查已应用于多智能体系统(multi-agents system,简称 MAS)的验证.MAS 是一种新型的智能系统设计和开发范式,由多个自主智能体组成,其目的是将大而复杂的系统划分为多个简单自主智能体、彼此互相通信和协调来完成系统功能. MAS 已被广泛应用于复杂系统的设计和开发,如自主航空器控制系统[4].在 MAS 中,每个智能体都有局部的状态(local state),其不确定性通过智能体的策略决定,全局状态(global state)由所有智能体的局部状态组成,其行为则由所有智能体共同决定.1997 年,Alur 等人提出了面向 MAS 的模型检查方法,其采用并发博弈结构(concurrent game structure,简称 CGS)进行建模,提出了交替时态逻辑(alternating-time temporal logic,简称 ATL 和 ATL*)用于描述系统规约,设计了相应的模型检查算法[5].ATL 和 ATL*是经典时态逻辑 CTL 和 CTL*的扩展,引入了联合模态操作(coalition modalities):⟨⟨A⟩⟩φ 和[[A]]φ,其中 A 是一个智能体集合.⟨⟨A⟩⟩φ 为真当且仅当 A 中的智能体有一个联合策略,无论其他智能体如何选择,目标公式 φ 都成立;而[[A]]φ 则是⟨⟨A⟩⟩φ 的对偶形式.在该框架中,所有智能体都可看到其他智能体所在的局部状态,同时可以记录所有历史状态.CGS 也可以通过解释系统(interpreted system,简称 IS)表达,其中每个智能体是一个 Kripke 结构,CGS 模型可以通过 Mealy 型或 Moore 型同步组合而获得[5].

在实际应用中,MAS 可能是一个分布式系统,每个智能体仅能观察到其自身的局部状态,而无法知道其他智能体所在的局部状态,即不完全信息(imperfect information);或由于资源问题,智能体无法记录所有历史状态,即不完全记忆(imperfect recall).为解决这个问题,Schobbens 利用 CGS 模型的语义刻画了智能体的能力,定义其相应的策略类型[6],并对 ATL 和 ATL*模型检查问题进行了研究.在不完全记忆场景下,每个智能体的决策仅依赖于系统的当前状态,反之依赖于历史状态有限路径;在不完全信息场景下,每个智能体在面对其不可区分的全局状态或历史状态有限路径时必须选择一样的策略.以下我们用 R 表示完全记忆, r 表示不完全记忆.类似的,我们用 I 表示完全信息, i 表示不完全信息.Schobbens 指出在 iR 场景下,ATL 模型检查问题是不可判定的,证明见[7].在其他场景下,模型 ATL 和 ATL*模型检查问题是可判定的,比如 ATL 和 ATL*模型检查在 ir 场景下分别可以在 P^{NP} 和 PSPACE 时间判定,详见[6].Jamroga 等人证明 ATL 模型检查问题在 ir 场景下是 P^{NP}-完备的,并设计实现了一个高效的算法[8,9].虽然在完全信息时,是否完全记忆对 ATL 模型检查没有任何影响,但是在不完全信息场景下或对 ATL*逻辑,是否完全记忆却具有重要的影响[23].除此之外,为了描述更多的系统规约,Chatterjee 等人提出了 LTL 逻辑的一阶量词扩展逻辑(Strategy Logic,简称 SL)[10,11,12],在 ATL、ATL*或 AMC 中引入知识操作等

[13,14,15,16,17].

虽然这些工作都考虑了智能体的不同能力,但仍然存在如下问题.考虑⟨⟨A⟩⟩φ的语义,在 IR 场景下,A 中的智能体和不在 A 中的智能体(用 $\overline{A}$ 表示)都具有相同的能力.但是在 Ir,iR 或 ir 场景时,已知工作中仅仅考虑了 A 中的智能体要符合 Ir,iR 或 ir 能力,而 $\overline{A}$ 中的智能体却采用 IR 策略.这样做的一个好处是:不管在何种场景下,⟨⟨∅⟩⟩φ 均可对应到 ATL/ATL*中的所有路径量词 ∀,[[∅]] 对应到存在路径量词 ∃.可见, A 和 $\overline{A}$ 的智能体拥有不同的能力.但是当联合模态操作出现嵌套时,例如公式⟨⟨A⟩⟩φ 包含一个子公式⟨⟨A'⟩⟩φ',且 $A \neq A'$ 时,某些智能体为了满足公式⟨⟨A⟩⟩φ 和⟨⟨A'⟩⟩φ',将会具有不同的策略类型,导致智能体策略类型的不一致性.这种将智能体的能力隐含于公式中的方式既不利于理解,也不利于应用.

为解决这个问题,有两种方案.第一种是在 MAS 中指明智能体所具备的能力;另一种是在逻辑公式中指明智能体所具备的能力.本文采用第一种方案,主要出于以下两方面的原因.1)智能体的策略类型往往取决于 MAS 本身,且不同智能体在同一 MAS 中具备不同的策略类型,这类系统称为异构多智能体系统(heterogeneous mutli-agent system). 同时一个 MAS 中智能体的策略能力一般不会随着验证规约不同而改变.2)第二种方案需要在规约中显示枚举所有智能体,且在嵌套联合模态操作中容易出现智能体策略类型前后不一致.

基于第一种方法,本文提出了一种解释系统的扩展,带类型解释系统(typed interpreted system,简称 TIS),其中每个智能体被赋予了 IR,Ir,iR 或 ir 中的一种策略类型,代表智能体在系统中的策略能力.在一个 TIS 中,智能体可以具备不同策略能力,因此可以对异构多智能体系统进行建模,也避免智能体策略类型的不一致性问题.本文根据 TIS 模型,定义了 ATL 的语义,并研究其模型检查问题.当 TIS 系统中的智能体被赋予 iR 能力,由于 ATL 模型检查在 iR 场景下是不可判定的[7],这将直接导致此情况下 ATL 在 TIS 上的模型检查问题是不可判定的.当 TIS 中智能体仅被赋予 Ir,IR 或 ir 能力,并且 ATL 公式中出现的智能体都不是 IR 策略类型时,ATL 模型检查问题是可判定的,并提出了一种复杂度为 EXPTIME 算法,设计实现了基于有序二叉决策图(ordered binary decision diagram, 简称 OBDD)的符号化模型检查工具 ShTMC,实验结果证明了 ShTMC 的可用性.

本文第 1 节首先介绍相关工作.第 2 节简介背景知识,包括解释系统和 ATL 逻辑等.第 3 节通过一个示例揭示经典语义的不足.第 4 节提出带类型解释系统,研究了 ATL 的语义,提出了模型检查算法.第 5 节介绍实验结果.第 6 节总结和展望.

1 相关工作

近几年,ATL/ATL*语义定义已经引起了国内外的关注.在经典 ATL/ATL*语义中,对于公式⟨⟨A⟩⟩φ,当 A 中智能体选择了某联合策略,对目标 φ 进行验证时,如果遇到其他联合模态操作,如⟨⟨A'⟩⟩φ'或[[A']]φ',所有智能体将忘记之前选择的策略,然后重新选择新的策略去满足 φ',即可撤销策略(revocable strategies).这种语义定义方式与博弈论的语义不一致.为此,Agotnes 等人提出了不可撤销的语义(irrevocable strategies)[18],即 A 中智能体确定了某一种联合策略后,即使遇到其他联合模态操作,A 中智能体还是必须遵照之前选择的策略进行.根据这种语义,Agotnes 等人给出了 ATL 在 IR 和 Ir 场景下的模型检查算法.

为了能在规约上对策略进行约束,Pinchinat 在 Dμ 逻辑中引入了决策操作[19],使得可以在规约中建立谓词和智能体策略间的联系,并对该逻辑的表达能力进行了研究.

在 Agotnes 等人工作基础之上[18],Brihaye 等人在 ATL/ATL*中引入了策略上下文(strategy contexts)和记忆约束(memory constraints)[19,20,21].策略上下文可以保存智能体选择的策略,从而解决经典语义中的策略不可撤销性问题,同时还能指定哪些智能体的策略是可撤销的.而记忆约束则限制了智能体能保存的信息量.在这种语义下,Brihaye 等人对 Ir 和 IR 场景下的模型检查问题和逻辑表达力进行了分析.

Goranko 等人研究了基于博弈论的 ATL^+的语义,该语义与经典的语义等价,并在此基础上提出了一种新的模型检查算法[22].除上述工作之外,为了平衡逻辑表达能力和模型检查算法复杂度,Wang 等人提出了 Strategy Logic 的两种子类:BSIL 逻辑[27]和 TCL 逻辑[29],并对这两种逻辑的可满足性和模型检查问题进行了研究.

本文研究的语义跟上述工作都不同,上述工作都假设在一个 MAS 中,智能体的策略类型是一样的或不在模态操作中出现的智能体都采用 IR 策略类型,而本文提出的语义却允许在同一 MAS 中,各智能体拥有不同的策

略类型,因此之前的模型检查算法均不适用于本文研究的语义.另外,本文的智能体策略类型是通过模型在语法层进行定义,而非以往工作通过语义定义.

2 背景知识

本节简单回顾 Kripke 结构,并发博弈结构,解释系统和交替时态逻辑 ATL.

2.1 Kripke结构

定义 1.假定 AP 为非空原子命题集合,Kripke 结构(Kripke structure)是一个四元组:

$$K \triangleq (Q,Q_0,\delta,L)$$

其中:

- Q是有限状态集合.
- $Q_0 \subseteq Q$ 是初始状态集合.
- $\delta: Q \to 2^Q$ 是状态迁移函数.
- $L:AP \to 2^Q$ 是系统的标签函数,对于任意状态 q 和原子命题 $p,q \in L(p)$ 表示 p 命题在状态 q 时为真.

K 的路径(path)是一个无限状态序列 $\pi = g_0g_1g_2\cdots$,其中对任意 $i \geq 0$,满足 $g_{i+1} \in \delta(g_i)$. Paths(g) 表示所有从状态 g 出发的路径集合.

2.2 并发博弈结构

并发博弈结构(concurrent game structure,简称 CGS)通过引入"智能体行为"这一概念对 Kripke 模型进行了扩展.对于 CGS,它的系统状态演变取决于系统中所有智能体的并发行为.

定义 2.假定 AP 为非空原子命题集合,CGS 是一个多元组:

$$G \triangleq (Q,Q_0,Agt,Ac,(\sim_i)_{i\in Agt},\lambda,\delta,L)$$

其中:

- Q是有限状态集合.
- $Q_0 \subseteq Q$ 是初始状态集合.
- $Agt = \{1,...,k\}$ 为有限非空智能体集合.
- Ac 是有限动作集合.
- $\sim_i \subseteq Q \times Q$ 是状态的等价关系, $g \sim_i g'$ 表示状态 g 和 g' 对于智能体 i 是不可区分的.
- $\lambda: Q \times Agt \to 2^{Ac}$ 是智能体的动作规范,对于一个给定状态 g ,$\lambda(g,i)$ 表示智能体 i 在状态 g 时所有可选择的动作集,且满足 $\lambda(g,i) = \lambda(g',i)$,若 $g \sim_i g'$ 成立.
- $\delta: Q \times Ac^{Agt} \to Q$ 是状态迁移函数满足:对于任意状态 g ,动作组 $\langle a_1,...a_k\rangle$,如果 $\delta(g,\langle a_1,...a_k\rangle)$ 有定义,则 $a_i \in \lambda(g,i)$.
- $L:AP \to 2^Q$ 是系统的标签函数,对于任意状态 q 和原子命题 $p,q \in L(p)$ 表示 p 命题在状态 q 时为真.

G 的路径(path)是一个无限状态序列 $\pi = g_0g_1g_2\cdots$,其中对任意 $i \geq 0$,存在动作组 $\alpha_i \in Ac^{Agt}$,满足 $g_{i+1}=\delta(g_i,\alpha_i)$.给定一条路径 $\pi = g_0g_1g_2\cdots$,路径的一个任意前缀称为有限路径.

2.3 解释系统

解释系统(interpreted system,简称 IS)是 Kripke 结构的另一种扩展[15],相比于 CGS 更适用于 MAS 的建模.因为 IS 的全局状态是由其内部所有智能体的局部状态组成的,而在 CGS 中,系统只有全局状态这一概念,因此 CGS 中无法直接观察单个智能体的局部状态迁移关系.当系统不满足规约时,也无法从反例中具体分析是否某个智能体导致规约不成立,除非有全局状态到局部状态的映射关系.鉴于 IS 的这一特性,本文将以 IS 为例进行扩展,但本文提出的思想同样适用于扩展 CGS.

定义 3.假定 Agt = {l,...,k} 为智能体集合,AP 为非空原子命题集合,IS 是一个多元组:

$$J \triangleq (\langle Q_i, Ac_i, \lambda_i, \delta_i, Q_i^0 \rangle_{i \in Agt}, L)$$

其中:

- Q_i 是智能体 i 的局部状态集合.所有智能体的局部状态集合的积 $Q_1 \times ... \times Q_k$ 为系统的全局状态集合,本文用 Q 表示系统的全局状态集合.对于任意一个全局状态 $g \in Q$, g(i) 表示在全局状态 g 中,智能体 i 的局部状态.
- Ac_i 是智能体 i 可选的非空有限动作集.所有智能体动作集合的积 $Ac = Ac_1 \times ... \times Ac_k$ 被称为系统的联合动作集.联合动作用 $\alpha = \langle a_1, ..., a_k \rangle \in Ac_1 \times ... \times Ac_k$ 表示, $\alpha(i)$ 表示智能体 i 的动作.
- $\lambda_i : Q_i \to 2^{Ac_i}$ 是智能体 i 的动作规范,给定局部状态 q, $\lambda_i(q)$ 表示智能体 i 在局部状态 q 时所有可选择的动作集,定义 $\lambda_i(g) = \lambda_i(g(i))$.这表明智能体的行为是受限的,可选动作与其当前所处的局部状态有关.对于一个给定的全局状态 g ,以 $\lambda(g)$ 表示集合 $\lambda_1(g) \times ... \times \lambda_k(g)$.
- $\delta_i : Q_i \times Ac \to Q_i$ 是智能体 i 的状态转移函数,给定局部状态 q 和联合动作 α 满足 $\alpha(i) \in \lambda_i(q)$, $\delta_i(q, \alpha)$ 是 i 在下一时刻的局部状态.定义 δ 为全局状态迁移关系,满足 $\delta((q_1, ..., q_k), \alpha) = (\delta_1(q_1, \alpha), ..., \delta_k(q_k, \alpha))$.
- $Q_i^0 \subseteq Q_i$ 是智能体 i 的初始状态集合,本文用 Q_0 表示全局初始状态集合 $Q_1^0 \times ... \times Q_k^0$.
- $L : AP \to 2^Q$ 是系统的标签函数,对于任意的全局状态 g 和原子命题 p, $g \in L(p)$ 表示 p 命题在全局状态 g 时为真.

从解释系统 J ,可以推导出 CGS 系统 $(Q, Q_0, Agt, Ac', (\sim_i)_{i \in Agt}, \lambda', \delta, L)$,其中 Q, Q_0, Agt, δ和L 如定义 3 所述, $Ac' = Ac_1 \cup ... \cup Ac_k$, λ' 定义为 $\lambda'(g, i) = \lambda_i(g(i))$, $\sim_i$ 定义为: $g \sim_i g'$ 当且仅当 g(i)=g'(i) .

J 的路径(path)是一个无限全局状态序列 $\pi = g_0 g_1 g_2 \cdots$,其中对任意 $m \geq 0$,存在一个联合动作 $\alpha_m \in Ac$,满足 $g_{m+1} = \delta(g_m, \alpha_m)$.给定一条路径 $\pi = g_0 g_1 g_2 \cdots$ 和 $j \geq 0$,以 $\pi(j)$ 表示序列中第 j 处的全局状态 g_j ,并用 $proj_i(\pi)$ 代表智能体 i 在 π 中的分量,即局部状态序列 $g_0(i) g_1(i) g_2(i) \cdots$.

有限路径 $\tau = g_0 g_1 \cdots g_n$ 是路径 $\pi = g_0 g_1 g_2 \cdots$ 的前缀.给定有限路径 $\tau = g_0 g_1 \cdots g_n$ 和 $0 \leq m \leq n$,以 $\tau(m)$ 表示序列中第 m 处的全局状态 g_m, $lst(\tau)$ 是 τ 的最后一个状态, $proj_i(\tau)$ 代表智能体 i 在 τ 中的分量,即局部状态序列 $g_0(i) g_1(i) \cdots g_n(i)$.

下文将以 Paths 表示所有可能的路径集,以 Trks 表示所有可能有限路径的集.对于一个给定的全局状态 g , 以 Paths(g) 和 Trks(g) 表示所有从 g 出发的路径和有限路径集合.

2.4 策略

在一个 IS 中,智能体 i 在某一状态下可能存在多个可以选择的动作,这种不确定性采用策略(strategy) θ_i 来决定.而策略根据智能体是否可以观察到其他智能体的状态和是否依赖于历史状态有不同的定义方式.本文采用文献[6]的定义方式,定义四种不同策略类型:IR,Ir,iR 和 ir,具体定义如下:

- **IR.** $\theta_i : Trks \to Ac_i$:对于任何一个智能体 i 以及任意一条有限路径 τ ,满足 $\theta_i(\tau) \in \lambda_i(lst(\tau))$,即策略函数 θ_i 为智能体 i 定义了从有限路径到可选动作的映射函数.
- **Ir.** $\theta_i : Trks \to Ac_i$:对两条有限路径 τ_1 和 τ_2 满足 $lst(\tau_1) = lst(\tau_2)$,那么智能体 i 将选择相同的动作,即 $\theta_i(\tau_1) = \theta_i(\tau_2)$.因此,智能体 i 选择的动作只依赖系统当前的状态,不依赖于历史有限路径.
- **iR.** $\theta_i : Trks \to Ac_i$:类似于 IR 策略,不同的是,对于两条有限路径 τ_1 和 τ_2 ,如果 $proj_i(\tau_1) = proj_i(\tau_2)$,则 $\theta_i(\tau_1) = \theta_i(\tau_2)$.直观地讲,iR 策略表示智能体 i 在任何两条无法区分的有限路径上都选择同一个动作.
- **ir.** $\theta_i : Trks \to Ac_i$:ir 融合了 iR 和 Ir 的所有特点,在 ir 中,如果 $lst(\tau_1)(i) = lst(\tau_2)(i)$,则 $\theta_i(\tau_1) = \theta_i(\tau_2)$.即智能体 i 仅能根据当前自身的局部状态来判断选择一个动作.

给定智能体集合 A, $\overline{A}$ 表示其补集,A 的联合 σ- 策略就是 A 中所有智能体的 σ- 策略构成的集合,即 $\theta_A = \{\theta_i \mid i \in A\}$,其中 $\sigma \in \{IR, Ir, iR, ir\}$,本文用 $\theta_A(i)$ 表示智能体 i 在 θ_A 中的策略.

给定智能体集 A,全局状态 g,以及联合 σ- 策略 θ_A ,以 $Out_\sigma(g, \theta_A)$ 表示 A 中的所有智能体严格按照策略 θ_A

选择动作,从初识状态 g 开始,系统所有可能路径的集合,即对于任意 $\pi \in Out_\sigma(g,\theta_A)$ 都有 $\pi(0)=g$,以及任意时刻 $j \geq 0$,存在联合动作 $\alpha \in \lambda(\pi(j))$,满足 $\pi(j+1)=\delta(\pi(j),\alpha)$ 和对于所有智能体 $i \in A$, $\alpha(i)=\theta_A(i)(\pi(0)...\pi(j))$ 成立.

2.5 交替时态逻辑ATL

交替时态逻辑(alternating-time temporal logic,简称 ATL)是 CTL 的一种扩展[5],将存在路径量词∃替换为联合模态操作〈〈A〉〉,其中 A 为一个智能体集合.

定义 4.ATL 的语法如下:

$$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle\langle A\rangle\rangle X\varphi \mid \langle\langle A\rangle\rangle G\varphi \mid \langle\langle A\rangle\rangle \varphi_1 U \varphi_2$$

其中 $A \subseteq Agt$.

定义 5.给定解释系统 $J=(\langle Q_i, Ac_i, \lambda_i, \delta_i, Q_i^0\rangle_{i\in Agt}, L)$, 全局状态 g 和 ATL 公式 φ,公式的可满足性 $J,g \models_\sigma \varphi$ 通过如下所示归纳定义:

- $J,g \models_\sigma p$ 当且仅当 $g \in L(p)$;
- $J,g \models_\sigma \neg\varphi$ 当且仅当 $J,g \not\models_\sigma \varphi$;
- $J,g \models_\sigma \varphi_1 \vee \varphi_2$ 当且仅当 $J,g \models_\sigma \varphi_1$ 或 $J,g \models_\sigma \varphi_2$;
- $J,g \models_\sigma \langle\langle A\rangle\rangle X\varphi$ 当且仅当存在一个联合 σ-策略 θ_A,对于所有路径 $\pi \in Out_\sigma(g,\theta_A)$ 都有 $J,\pi(1) \models_\sigma \varphi$;
- $J,g \models_\sigma \langle\langle A\rangle\rangle G\varphi$ 当且仅当存在一个联合 σ-策略 θ_A,对于所有路径 $\pi \in Out_\sigma(g,\theta_A)$,在任何时刻 $j \geq 0$,都有 $J,\pi(j) \models_\sigma \varphi$;
- $J,g \models_\sigma \langle\langle A\rangle\rangle \varphi_1 U \varphi_2$ 当且仅当存在一个联合 σ-策略 θ_A,对于所有路径 $\pi \in Out_\sigma(g,\theta_A)$,都存在一个时刻 $j \geq 0$,使得 $J,\pi(j) \models_\sigma \varphi_2$ 成立,并且对于所有时刻 $0 \leq i < j$ 都有 $J,\pi(i) \models_\sigma \varphi_1$ 成立.

解释系统 J 满足公式 φ 当且仅当所有 J 中的全局初始状态都满足 φ.类似的,可以定义 ATL 在 CGS 系统上的语义,详见[5].

给定一个公式 φ,在某语义下,所有智能体采用同一个策略类型,即称为智能体的策略类型是一致的,反之为不一致的.考虑公式〈〈{1,2,3}〉〉(p U〈〈{1,2}〉〉Xφ),在 IR 语义时,是一致的.而在其他语义时,由于智能体 3 的策略类型在满足 p U〈〈{1,2}〉〉Xφ 和 φ 时可能会不同,因此是不一致的.

计算树逻辑(computation tree logic,简称 CTL)是 ATL 逻辑的一种特殊子类,其中所有〈〈A〉〉φ 公式中的 A 必须是空集 ∅,即〈〈∅〉〉表示所有路径量词.CTL 在 Kripke 结构的语义见[1].

定理 1.给定 IS 系统 $J=(\langle Q_i, Ac_i, \lambda_i, \delta_i, Q_i^0\rangle_{i\in Agt}, L)$,全局状态 g 和 ATL 公式 φ,以下结论成立:

(1) $J,g \models_{iR} \varphi$ 问题是不可判定的.

(2) $J,g \models_{IR} \varphi$ 和 $J,g \models_{Ir} \varphi$ 问题是 PTIME-完备的[5,23].

(3) $J,g \models_{IR} \varphi$ 当且仅当 $J,g \models_{Ir} \varphi$ [23].

(4) $J,g \models_{ir} \varphi$ 问题是 P^{NP}-完备的[6,8].

证明:Dima and Tiplea 证明了 ATL 在 CGSs 上的模型检查问题在 iR 场景下是不可判定的[7],其将图灵机停机问题规约到一个仅包含三个智能体的 CGS 系统 G 满足 ATL 公式〈〈{1,2}〉〉G ok 的问题.因此只需要将该 CGS 系统转化到一个 IS 系统 J,使得 $J,g \models_{iR}$ 〈〈{1,2}〉〉G ok 当且仅当 G 中的状态 g 在 iR 场景满足〈〈{1,2}〉〉G ok.

考虑 CGS 系统 $G=(Q,Q_0,Agt,Ac,(\sim_i)_{i\in Agt},\lambda,\delta,L)$ 对于任意智能体 i,假设 $Q_i=\{Q_i^1,...,Q_i^{n_i}\}$ 为关系 $\sim_i$ 在 Q 上的等价类集,以 $|g|_i$ 表示状态 g 在 Q_i 中的等价类.对于公式〈〈{1,2}〉〉G ok 而言,智能体 3 的等价类 Q_3 不影响其可满足性,因此 $\sim_3$ 可以替换为最小等价关系,即 $g \sim_3 g'$ 当且仅当 g=g'.本文假设 $\sim_3$ 为最小等价关系.

给定状态 g 和动作组 $\alpha=\langle\alpha_1,\alpha_2,\alpha_3\rangle$,以 α_g 表示动作组 $\alpha=\langle\alpha_1,\alpha_2,(\alpha_3,g)\rangle$.

定义 IS 系统 $J=(\langle Q_i, Ac_i, \lambda_i, \delta_i, Q_i^0\rangle_{i\in Agt}, L)$ 其中:

- $Ac_1 = Ac_2 = Ac, Ac_3 = Ac \times Q$;

- 对于任意的 i, $\delta_i(|g|_i,\alpha_g)=|g'|_i$ 当且仅当 $\delta(g,\alpha)=g'$;
- $\lambda_i(|g'|_i)=\{\alpha_g|\alpha\in\lambda(g,i),g\in|g'|_i\}$;
- $Q_i^0=\{|g|_i|g\in Q\}$;

对于公式〈〈{1,2}〉〉G ok ,只有智能体 1 和 2 需要考虑联合 iR-策略,而智能体 3 是采用 IR-策略.根据 ATL 语义, $J,g|=_{iR}$ 〈〈{1,2}〉〉G ok 当且仅当 G 的状态 g 在 iR 场景满足〈〈{1,2}〉〉G ok .

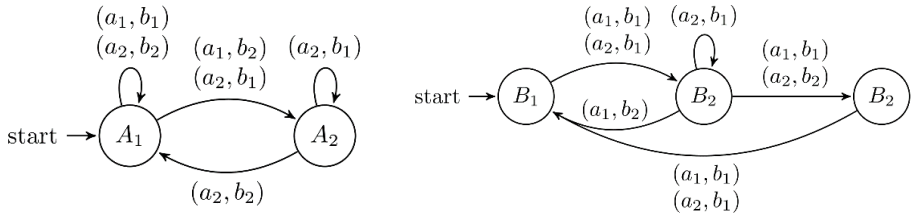

Fig.1 Local transition graphs of A and B

图 1 智能体 A 和 B 的局部状态转移图

2.6 示例

考虑系统模型 $J=(\langle Q_i,Ac_i,\lambda_i,\delta_i,Q_i^0\rangle_{i\in Agt},L)$,其中:

- $Agt=\{A,B\}$;
- $Q_A=\{A_1,A_2\},Q_B=\{B_1,B_2,B_3\}$;
- $Ac_A=\{a_1,a_2\},Ac_B=\{b_1,b_2\}$;
- δ_A 和 δ_B 如图 1 所示,节点表示局部状态,边表示状态迁移关系,边上的标识是智能体的联合动作;
- $\lambda_A=\{A_1:\{a_1,a_2\},A_2:\{a_2\}\},\lambda_B=\{B_1:\{b_1\},B_2:\{b_1,b_2\},B_3:\{b_1\}\}$;
- $Q_A^0=\{A_1\},Q_B^0=\{B_1\}$;
- $L(eva)=\{(A_2,B_2)\}$;

示例的全局状态转移关系见图 2.

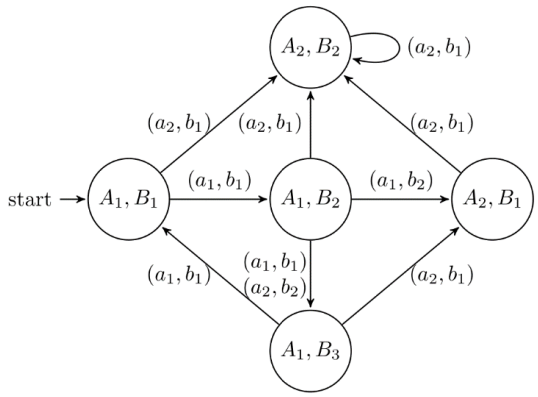

Fig.2Transitionrelation of system

图 2 全局状态迁移关系

考虑 ATL 公式: $\varphi=$〈〈{B}〉〉True U eva .不难发现若:智能体 A 在状态(A_1,B_1)时选择动作 a_1,系统将进入状态(A_1,B_2);在状态(A_1,B_2)时,如果 B 选择动作 b_1,A 选择动作 a_1;如果 B 选择动作 b_2,A 选择动作 a_2,系统将进入状态(A_1,B_3);若 A 继续选择动作 a_1,系统重回状态(A_1,B_1).如此反复,系统将永远不会到达状态(A_2,B_2).因此,对于任意的 $\sigma\in\{IR,Ir,iR,ir\}$, $J,(A_1,B_1)|=_\sigma\varphi$ 都不成立.

但是,在实际应用中,比如在无线传感器网络中,每个节点是一个智能体,其存储资源非常有限,同时为了降

低能耗,不可能时刻跟其他智能体保持通信来获取其他智能体的局部状态.因此,在上述例子中,如果将 A 的策略类型限制为 ir,在状态(A_1,B_2)时,A 只能选择动作 a_1;如果 B 选择动作 b_2,系统最终将到达(A_2,B_2).因此,智能体的策略类型对公式可满足性的结果有直接的影响.而在经典的语义下,无法对此类规约进行描述和验证,同时 CGS 和 IS 都无法对这类异构多智能体系统建模.针对这个问题,本文将在解释系统中引入智能体策略类型函数来描述智能体的策略类型,从而验证上述类型的规约.

3 基于策略类型的解释系统

通过上面的讨论可知,传统的多智能体系统验证框架在语义表述方面仍然存在不足,因此在这一节,本文将提出基于策略类型的解释系统,为每个智能体引入专属的策略类型.

3.1 TIS模型

定义 6.基于策略类型的解释系统(typed interpreted system,简称 TIS)是一个二元组

$$T \triangleq (J,\Lambda)$$

其中:

- $J=(\langle Q_i,Ac_i,\lambda_i,\delta_i,Q_i^0\rangle_{i\in Agt},L)$ 是一个解释系统,
- $\Lambda: Agt \to \{IR,Ir,iR,ir\}$ 是一个映射函数,为每个智能体关联一种策略类型.

TIS 依旧沿用传统解释系统中对路径和有限路径的定义,但是需要重新定义策略.与已有工作通过语义来定义智能体的策略不同,在 TIS 模型中,智能体的策略类型由 TIS 模型给出,如 Λ(i) 表示智能体 i 只能采用 Λ(i)-策略.

给定一个智能体集合A,A 的联合策略就是 A 中所有智能体的策略构成的集合,即

$$\theta_A \triangleq \{\theta_i \mid i\in A,\theta_i\text{是}\Lambda(i)\text{-策略}\}.$$

本文用 $\theta_A(i)$ 表示智能体 i 在 θ_A 中的策略,以 S_A 表示 A 所有可能的联合策略集合.已知智能体集合 A 和初始状态 g,两个联合策略 θ_A 和 $\theta_{\bar{A}}$ 唯一地确定了一条路径 $Path(\theta_A,\theta_{\bar{A}}) \triangleq \pi$,满足 $\pi(0)=g$,以及任意一个时刻 j,都存在联合动作 $\alpha\in\lambda(\pi(j))$,使得 $\pi(j+1)=\delta(\pi(j),\alpha)$,其中对于任意 $i\in A,\alpha(i)=\theta_A(i)(\pi(0)...\pi(j))$;任意 $i\in\bar{A},\alpha(i)=\theta_{\bar{A}}(i)(\pi(0)...\pi(j))$.给定智能体集合 A,初始状态 g,一个联合策略 θ_A,定义

$$Out(g,\theta_A) \triangleq \bigcup_{\theta_{\bar{A}}\in S_{\bar{A}}} Path(\theta_A,\theta_{\bar{A}}).$$

事实 1.给定 TIS T=(J,Λ) 和智能体集合 A,对于任意 $\sigma\in\{IR,Ir,iR,ir\}$,如果所有 $i\in A,\Lambda(i)=\sigma$,则以下结论成立:

1) θ_A 是 J 中 A 的一个联合 σ-策略当且仅当 θ_A 是 T 中 A 的联合策略;
2) 对于 T 中 A 的任意一个联合策略 θ_A,$Out(g,\theta_A)\subseteq Out_\sigma(g,\theta_A)$;
3) 当所有 $i\in\bar{A}$ 满足 Λ(i)=IR 时,对于任意一个 T 中 A 的联合策略 θ_A,$Out(g,\theta_A)\equiv Out_\sigma(g,\theta_A)$.

证明:根据联合 σ-策略和联合策略的定义,1)显而易见.

$Out(g,\theta_A)\subseteq Out_\sigma(g,\theta_A)$:对于任意路径 $\pi\in Out(g,\theta_A)$,证明 $\pi\in Out_\sigma(g,\theta_A)$ 成立即可.假设一条路径 $\pi=g_0g_1...$,其中 $g_0=g$,即对于智能体集 $\bar{A}$,始终存在联合策略 $\theta_{\bar{A}}=\{\theta_i|i\in\bar{A}\}\in S_{\bar{A}}$,任意时刻 $m\geq 0$,$g_{m+1}=\delta(g_m,\alpha_m)$ 成立,其中 α_m 是所有智能体的联合动作,即对于智能体 $i\in A$,有 $\alpha_m(i)=\theta_A(i)(g_0...g_m)$ 成立;对于智能体 $i\in\bar{A}$,有 $\alpha_m(i)=\theta_{\bar{A}}(i)(g_0...g_m)$ 成立.显然通过以上方法找到的路径,一定存在于 $Out_\sigma(g,\theta_A)$ 中.因此 2)成立.

$Out(g,\theta_A)\supseteq Out_\sigma(g,\theta_A)$:证明对于任意路径 $\pi\in Out_\sigma(g,\theta_A)$ 都有 $\pi\in Out(g,\theta_A)$ 成立.设有条路径 $\pi=g_0g_1...$,其中 $g_0=g$,且在任意 $m\geq 0$,$g_{m+1}=\delta(g_m,\alpha_m)$ 成立,同时对于智能体 $i\in A$,$\alpha_m(i)=\theta_{\bar{A}}(i)(g_0...g_m)$ 成立.此时,在任意 $m\geq 0$,智能体 $i\in\bar{A}$,我们令 $\theta_{\bar{A}}(i)(g_0...g_m)=\alpha_m(i)$.对于智能体 $i\in\bar{A}$,由于 Λ(i)=IR,则有 $\theta_{\bar{A}}\in S_{\bar{A}}$,使得 $\pi\in Out(g,\theta_A)$ 成立.综上 3)成立.

定义 6.给定一个 TIS T,全局状态 g 和 ATL 公式 φ,则可满足性 T,g |= φ 通过如下归纳定义:

- $T,g \models p$ 当且仅当 $g \in L(p)$;
- $T,g \models \neg\varphi$ 当且仅当 $T,g \not\models \varphi$;
- $T,g \models \varphi_1 \vee \varphi_2$ 当且仅当 $T,g \models \varphi_1$ 或 $T,g \models \varphi_2$;
- $T,g \models \langle\langle A\rangle\rangle X\varphi$ 当且仅当存在一个联合策略 θ_A,对于任意路径 $\pi \in Out(g,\theta_A)$ 都有 $T,\pi(1) \models \varphi$;
- $T,g \models \langle\langle A\rangle\rangle G\varphi$ 当且仅当存在一个联合策略 θ_A,对于任意路径 $\pi \in Out(g,\theta_A)$,在任何时刻 $j \geq 0$ 都有 $T,\pi(j) \models \varphi$;
- $T,g \models \langle\langle A\rangle\rangle \varphi_1 U \varphi_2$ 当且仅当存在一个联合策略 θ_A,对于所有路径 $\pi \in Out(g,\theta_A)$,都存在一个时刻 $j \geq 0$,使得 $T,\pi(j) \models \varphi_2$ 成立,并且对于所有时刻 $0 \leq i < j$ 都有 $T,\pi(i) \models \varphi_1$.

定理 2.ATL 在 TIS 上的模型检查问题是不可判定的.

证明:根据定理 1 的证明,可以把图灵机停机问题规约到一个 IS J 是否满足 $\langle\langle\{1,2\}\rangle\rangle G\ ok$ 的问题上.而从 J 模型可以构造成一个 TIS 系统 T=(J,Λ),其中 Λ 满足 $\Lambda(1)=\Lambda(2)=ir, \Lambda(3)=IR$,从而将图灵机停机问题规约到 ATL 在 TIS 模型检查问题.

由于定理 2,只需要考虑 TIS 系统 T=(J,Λ) 满足 $\Lambda: Agt \to \{IR,Ir,ir\}$ 的模型检查问题.本文研究满足 $\Lambda: Agt \to \{IR,Ir,ir\}$ 的 TIS 模型检查问题,但是 ATL 公式中出现的智能体必须是 Ir 或 ir 类型,即所有 ATL 公式中出现的 $\langle\langle A\rangle\rangle\varphi$ 公式满足:如果 $i \in A$,则 $\Lambda(i) \in \{Ir,ir\}$.对于一般情况,目前作者还没有解决方案,留待未来解决.

3.2 模型检查算法

对于任何一个智能体,如果其策略类型为 Ir 或 ir,那么其策略数量是有限的.因此可以通过不确定地猜测每个智能体的策略,然后再验证猜测是否正确.给定 TIS 系统 T 和 ATL 公式 φ 并且公式 φ 中出现的智能体是 Ir 或 ir 类型,$\|\varphi\|_T$ 表示所有满足公式 φ 的状态集合.在给出具体算法前,先引入一些基本定义.

给定智能体集合 A 和策略类型函数 $\Lambda: Agt \to \{IR,Ir,ir\}$,$A_r^\Lambda$ 表示智能体集合 A 的子集并满足:$i \in A_r^\Lambda$ 当且仅当 $\Lambda(i) \in \{Ir,ir\}$. F_A^Λ 表示 $f: A_r^\Lambda \times Q \to Ac$ 函数集合并满足:对任意 $i \in A_r^\Lambda$,任意状态 $g \in Q$,

1) $f(i,g) \in \lambda_i(g(i))$;
2) 如果 $\Lambda(i)=ir$,则对任意状态 $g' \in Q$,若 $g(i)=g'(i)$ 成立,则 $f(i,g)=f(i,g')$.

给定两个函数 $f \in F_A^\Lambda, \overline{f} \in \overline{F}_A^\Lambda$ 和 TIS 系统 T,$T(f,\overline{f})$ 表示一个 Kripke 结构 (Q,Q_0,δ',L),其中 δ' 定义为:$q' \in \delta'(q)$ 当且仅当 $q=(q_1,...,q_k), q'=(q'_1,...,q'_k)$,存在一个联合动作 $\alpha \in Ac$ 满足如下条件:

1) $\delta(q,\alpha)=q'$;
2) 对任意 $i \in Agt$,如果 $i \in A_r^\Lambda$,则 $f(i,q)=\alpha(i)$;如果 $i \in \overline{A}_r^\Lambda$,则 $\overline{f}(i,q)=\alpha(i)$.

算法 1.模型检查算法

输入: ATL 公式 φ,TIS 系统 $T=((\langle Q_i,Ac_i,\lambda_i,\delta_i,Q_i^0\rangle_{i\in Agt},L),\Lambda)$

输出: $\|\varphi\|_T$

MC(T, φ)

1.**begin**

2.**switch** φ **do**

3. **case** φ=p : **return** L(p);
4. **case** φ=¬φ' **:return**Q/MC(T, φ');
5. **case** $\varphi=\varphi_1 \vee \varphi_2$ **: return**$MC(T,\varphi_1) \cup MC(T,\varphi_2)$;
6. **case** $\varphi=\langle\langle A\rangle\rangle\varphi'$:
7. **for each** subformula $\phi=\langle\langle A'\rangle\rangle\varphi''$ in φ' **do**
8. Replace ϕ by a fresh atomic proposition p_ϕ in φ
9. Let $L(p_\phi)=MC(T,\phi)$;
10. **end for**
11. S ≔ ∅;

12. **for each** $f \in F_A^\Lambda$ **do**
13. $S := S \cup (\bigcap_{\bar{f} \notin F_{\bar{A}}^\Lambda} MC(T(f,\bar{f}),\langle\langle\varnothing\rangle\rangle\phi'))$;
14. **end for**
15. **return** S;
16. **end switch**
17. **end**

$\|\varphi\|_T$ 计算方法如算法 1 所示,其中假设对于一个给定的 Kripke 结构和 CTL 公式 φ,MC(K,φ) 返回满足公式 φ 的状态集合,该问题可以在多项式时间内解决[1].具体来说,输入一个 ATL 公式 φ 和一个 TIS 系统 T,通过公式的结构归纳法计算满足子公式的状态集合.第 2 行~第 5 行显而易见,难点在公式 $\varphi=\langle\langle A\rangle\rangle\varphi'$.由于 φ' 中可能有嵌套形如 $\phi=\langle\langle A'\rangle\rangle\varphi''$ 的子公式,对于这种情况,采用递归调用,先计算满足这些公式的状态集,然后把这些公式替换为一些新的原子命题,并更新 L 函数(第 7 行~第 10 行).根据假设对任意 $i \in A$, $\Lambda(i) \in \{Ir,ir\}$ 成立,因此 A 中智能体的策略数量是有限的,可以通过枚举法验证每一条可能的策略.对每种 A 中智能体策略的策略组合 $f \in F_A^\Lambda$,考虑所有 $\bar{A}$ 的策略.对于 $\bar{A}$ 策略类型为 Ir 或 ir 智能体,同理枚举所有可能的策略组合.而策略类型为 IR 的策略数量是无限的,无法通过枚举进行处理.因此将问题规约到 CTL 在 Kripke 结构上的模型检查问题.其主要思想为:$g \in \|\langle\langle A\rangle\rangle\phi'\|_T$ 当且仅当存在一个联合策略 θ_A (等价于第 12 行的 f),任意路径 $\pi \in Out(g,\theta_A)$ 都满足 φ';即当且仅当存在一个联合策略 $\theta_{\bar{A}_r^\Lambda}$,对任意联合策略 $\theta_{\bar{A}_r^\Lambda}$ (等价于第 13 行的 $\bar{f}$),任意路径 $\pi \in Out(g,\theta_A \oplus \theta_{\bar{A}_r^\Lambda})$ 都满足 φ',其中 $\theta_A \oplus \theta_{\bar{A}_r^\Lambda}$ 是一个联合策略,满足:对于任意 $i \in Agt$,任意状态 $g \in Q$,

$$(\theta_A \oplus \theta_{\bar{A}_r^\Lambda})(i)(g) = \theta_A(i)(g), \text{if } i \in A$$
$$(\theta_A \oplus \theta_{\bar{A}_r^\Lambda})(i)(g) = \theta_{\bar{A}_r^\Lambda}(i)(g), \text{if } i \in \bar{A}_r^\Lambda$$

根据 $T(f,\bar{f})$ 定义可见,对于任意 g, $T(f,\bar{f})$ 中的 Paths(g) 等价于 $Out(g,\theta_A \oplus \theta_{\bar{A}_r^\Lambda})$.

定理 3.给定一个 TIS $T=(J,\Lambda)$ 满足 $\Lambda: Agt \to \{IR,Ir,ir\}$ 时,对于所有 ATL 公式,其中出现的智能体 i 的策略类型 $\Lambda(i) \in \{Ir,ir\}$ 满足时,模型检查问题可在 EXPTIME 内判断.

证明:正确性通过 ATL 结构归纳方法证明,对于 $p,\neg\phi'$和$\varphi_1 \vee \varphi_2$,结论显然成立,因此仅需要考虑 $\langle\langle A\rangle\rangle\phi'$.由于 A 中的智能体智能采用 ir 或 Ir 策略,因此 F_A^Λ 中的函数一一对应到 S_A 中的联合策略 θ_A.同理,$F_{\bar{A}}^\Lambda$ 中的函数一一对应到 $S_{\bar{A}}$ 中的联合策略 $\theta_{\bar{A}}$.在 $T(f,\bar{f})$ 中 Paths(g) 与 T 中在给定 θ_A 和 $\theta_{\bar{A}}$ 时所有从 g 出发的路径 $Out(g,\theta_A \oplus \theta_{\bar{A}_r^\Lambda})$ 等价.

$g \in \bigcap_{\bar{f} \notin F_{\bar{A}}^\Lambda} MC(T(f,\bar{f}),\langle\langle\varnothing\rangle\rangle\phi')$ 当且仅当所有 $\pi \in Out(g,\theta_A)$ 满足公式 φ'.因此正确性成立.

复杂度:CTL 在 Kripke 结构上的模型检查问题可在多项式时间内解决,算法 1 中 6-14 行内需要考虑两个递归,而 F_A^Λ 和 $F_{\bar{A}}^\Lambda$ 中函数数量最多为 $|Ac|^{|Q|\times|Agt|}$,因此算法 1 可以在 EXPTIME 内终止.

备注 1:该模型检查算法同样适用于 ATL*逻辑,只需要对 MC(K,φ) 调用 CTL*在 Kripke 结构的模型检查算法即可.不仅如此,该算法也可以很容易地扩展"知识"操作,如 $K_i\varphi$ 表示智能体 i 知道 φ 为真.

4 ShTMC 模型检查工具及实验

根据上一节介绍的算法,本文在开源软件 MCMAS[24,25]基础上设计实现了基于 OBDD 的符号化模型检查工具 ShTMC.MCMAS 采用解释系统编程语言(interpreted systems programming language,简称 ISPL)描述多个智能体的状态转移,支持多种包括 CTL、ATL 逻辑的基于 OBDD 的符号化模型检查.为便于系统建模,MCMAS 中可以定义一种特殊的智能体,叫环境(Environment).环境中定义的状态信息可以全部或部分被其他智能体观察到,这样可以减少智能体间的同步和通信.本文在 ISPL 语言中增加智能体策略类型描述语法用于 TIS 建模,同时继承了环境智能体这一特性.

为了验证本文所提出方法的性能,以匿名协议——DCP(Dining Cryptographers Protocol)及变种为例,在

Cygwin 平台上进行实验.DCP 是一个基于数学不可解特性的基础安全匿名通信协议,其主要特点是通过提供匿名信息服务来避免恶意攻击.具体描述如下:该协议中有 k 个密码员一起就餐,餐费是匿名者支付的,可能是其中一个密码员付的餐费,或者是他们的雇主付的餐费.密码员之间尊重彼此支付餐费的隐私,但是他们希望知道是不是雇主付了餐费.为解决这个问题,他们制定了如下的协议:每个人都抛掷一个质地均匀的硬币,抛掷结果能且仅能被自己和坐在自己右边的人知晓.然后,每个人同时大声告知所有人,自己抛掷的硬币和坐在自己左边的人抛掷的硬币正反面是否一致.如果某个密码员就是匿名支付者,那么他所告知的硬币正反面是否一致的这个信息的就要和他亲眼看到的事实相反(即两个硬币正反面一致就要说成不一致;正反面不一致就要说成一致).此时,如果有奇数个人声明“硬币正反面不一致”,那么这次晚餐是由其中一个密码员支付的,但这位匿名密码员谁,其他密码员是不清楚的;反之,如果声明“硬币正反名不一致”的人数是偶数,那么这次餐费则是由他们的雇主支付的.

在这个例子中,每个密码员可以建模为一个智能体,其局部状态信息包括该密码员是否支付餐费(即:paid,not-paid)和是否看到硬币正反面一致(即:see-equal,see-different 和初始值 empty),其是否声明看到币正反面一致则用动作刻画(即:say-equal,say-different 和初始值 none).因此,密码员的局部状态数量为 6 个.另外有一个环境智能体用于建模每个密码员硬币投掷的正反面(即 0 和 1)情况,其局部状态数量为 8 个.全局状态数量为 8×6^k.但在这个系统中并不是所有全局状态都可达,比如在 k=3 时,可达的全局状态数量只有 96 个.

针对 DCP 模型,本文验证了 ATL 公式:<agent1>F(odd -> (agent1paid or agent2paid or agent3paid)).该公式是为分析 ShTMC 工具的性能随意构造的,其中 agent1 代表密码员 1,agent1paid 表示密码员 1 支付了餐费,agent2paid 和 agent3paid 类似.Odd 表示有奇数个人声明“硬币正反面不一致”.

下表展示了 ShTMC 模型检查工具对 DCP 系统的验证结果,其中第一列给出了密码员数量,m(n)表示一共有 m 个密码员,其前 n 个密码员是 ir 策略类型,其他则是 IR 策略类型,环境智能体也采用 IR 策略类型;第二列给出了系统中可达状态数量;第三列是 BDD 中布尔变量数;第四列给出了执行时间,---表示时间超过 40 分钟;第五列给出了 BDD 所使用的内存.BDD 变量顺序按照 MCMAS 默认排序.

Table 1Experimental Results

表 1 实验结果

智能体数量	可达状态数量	组合策略数量	BDD 变量数	执行时间(秒)	消耗内存(兆)
3 (0)	128	1	35	<0.01	10.20
3 (1)	128	16	35	0.013	10.48
3 (2)	128	256	35	0.076	11.22
3 (3)	96	4096	35	1.511	11.33
4 (0)	320	1	45	<0.01	10.47
4 (1)	320	16	45	0.022	11.17
4 (2)	320	256	45	0.181	11.52
4 (3)	320	4096	45	2.706	12.45
4 (4)	240	65536	45	73.133	13.53
5 (0)	768	1	55	0.031	10.40
5 (1)	768	16	55	0.037	10.55
5 (2)	768	256	55	0.152	11.43
5 (3)	768	4096	55	3.599	12.97
5 (4)	768	65536	55	103.88	15.43
5 (5)	576	1048576	55	2016.15	130.85
6 (0)	1792	1	65	0.034	10.76
6 (1)	1792	16	65	0.041	10.80
6 (2)	1792	256	65	0.155	11.73
6 (3)	1792	4096	65	5.574	17.95
6 (4)	1792	65536	65	151.357	33.68
6 (5)	1792	1048576	65	2270.37	220.71
6 (6)	1344	16777216	65	---	---

从表 1 可见,在密码员数量为 m 时,如果所有密码员都采用 ir 策略(即 n=m),那么可达状态数量会比 m>n>0 时少.这是因为当所有智能体都是 ir 策略时,智能体将无法区分全局状态故而只能选择同一个动作,并导致了系统中的某些状态不可达.当有一个或多个智能体不是 ir 时,原本只能选择同一动作的智能体可以选择更多动作,此时这些不可达的状态将可达.另一方面,在相同密码员数量时,随着采用 ir 策略的智能体数量增加,集合 F_A^{Λ} 会

变大,直接导致算法 1 需要更多的运行时间.虽然在 m>6 时,系统运行时间较多,但 BDD 的变量数目不大,说明 ShTMC 可以验证规模较大的系统.

5 结语与展望

针对以往工作在多智能体系统中智能体策略能力建模的不足,本文提出了带策略类型的解释系统作为异构多智能体系统模型,该系统模型允许各智能体有其独立的策略能力.以 ATL 逻辑为例,本文研究了 ATL 逻辑在带策略类型的解释系统下的语义和模型检查问题.由于模型表达能力过强,ATL 模型检查算法问题不可判定.因此本文考虑一种特殊的带策略类型的解释系统,其中在 ATL 公式联合模态操作中出现的智能体可以是 Ir 或 ir 策略类型,而其他智能体可以是 IR,ir 或者 Ir 策略类型,提出了一个 EXPTIME 复杂度的模型检查算法,设计并实现了工具原型 ShTMC.

本文仅仅是针对多智能体系统中智能体策略能力建模的不足问题的初步工作,尚有很多问题有待后续深入研究.1)其他表达力更强的 ATL 逻辑的扩展,如 ATL*,Strategy Logic 的语义和模型检查问题;2)其他可能的策略类型,如可撤销和可删除策略类型等.

References:

[1] Clarke EM,Grumberg O, Peled DA.Model Checking.Cambridge:MIT Press,1999.

[2] Clarke EM, Emerson EA.Design and Synthesis of Synchronization Skeletons UsingBranching-Time Temporal Logic.In: Proceedings ofWorkshop on Logics of Programs. Berlin: Springer,1981. 52-71.

[3] Kupferman O,Vardi MY.Module Checking.In: Proceedings of the 8th International ConferenceonComputer Aided Verification.Berlin: Springer, 1996. 75–86.

[4] MuscettolaN,NayakPP,PellB,WilliamsBC.Remote agent: To boldly go where no AI system has gone before.Artificial Intelligence, 1998,103(1-2):5–47.

[5] AlurR,Henzinger TA,Kupferman O. Alternating-Time Temporal Logic. Journal of the ACM,2002, 49(5): 672-713.

[6] Schobbens P.Alternating-time logic with imperfect recall. Electronic Notes in Theoretical Computer Science, 2004,85(2):82–93.

[7] DimaC, Tiplea FL.Model-checking ATL under imperfect information and perfect recall semantics is undecidable.CoRR,abs/1102.4225,2011.

[8] Jamroga W, Dix J. Model checking abilities under incomplete information is indeed Delta2-complete. In: Proceedings of the 4th European Workshop on Multi-Agent System. CEUR-WS, 2006.

[9] Pilecki J,Bednarczyk MA, Jamroga W. Model checking properties of multi-agent systems with imperfect information and imperfect recall.In: Proceedings of the 7th International Conference on Intelligent Systems. Berlin: Springer, 2014. 415–426.

[10] Chatterjee K,Henzinger TA,Piterman N. Strategy Logic.In: Proceedings of the 18th International Conference on Concurrency Theory, Berlin: Springer, 2007. 59–73.

[11] Mogavero F,Murano A,Vardi MY.Reasoning About Strategies. In: Proceedings of Annual Conference on Foundations of Software Technology and Theoretical Computer Science,Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.133-144.

[12] Mogavero F, Murano A, Vardi MY. Reasoning About Strategies: On the Model-Checking Problem.ACM Trans. On Computational Logic,2014, 15(4): 34:1-34:47.

[13] Jamroga W.Some remarks on alternating temporal epistemic logic.In: Proceedings of Formal Approaches to Multi-Agent Systems,2003. 133–140.

[14] Hoek W, Wooldridge M.Cooperation,knowledge,and time: Alternating-time temporal epistemic logic and its applications.Studia Logica,2003, 75(1):125–157.

[15] Lomuscio A, Raimondi F.Model checking knowledge,strategies,and games in multi-agent systems.In: Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems. New York: ACM, 2006. 161-168.

[16] Hoek W,Wooldridge M.Tractable multiagent planning for epistemic goals.In: Proceedings of the First International Joint

Conference on Autonomous Agents and Multiagent Systems.New York: ACM, 2002. 1167–1174.

[17] Bulling N,Jamroga W.Alternating epistemic mu-calculus.In: Proceedings of the 22nd International Joint Conferenceon Artificial Intelligence,Austin: AAAI, 2011. 109–114.

[18] Agotnes T,Goranko V,Jamroga W.Alternating-time temporal logics with irrevocable strategies.In: Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge. New York: ACM,2007.15–24.

[19] Pinchinat S.A generic constructive solution for concurrent games with expressive constraints on strategies.In: Proceedings of the 5th International Symposium on Automated Technology for Verification and Analysis,Berlin: Springer, 2007. 253–267.

[20] Lopes ADC,Laroussinie F,Markey N.ATL with Strategy Contexts: Expressiveness and Model Checking.In: Proceedings of the IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010. 120-132.

[21] Laroussinie F, Markey N.Augmenting ATL with strategy contexts.Information and Computation, 2015, 245: 98-123.

[22] Goranko V, Kuusisto A,Rönnholm R.Game-Theoretic Semantics for ATL+ with Applications to Model Checking.In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems,New York: ACM, 2017. 1277-1285.

[23] JamrogaW.Logical Methods for Specification and Verification of Multi-Agent Systems.Warszawa: ICS PAS Publishing House, 2015.

[24] Lomuscio A,Qu H,RaimondiF.MCMAS: A model checker for the verification of multi-agent systems.In: Proceedingsof the21st International Conference on Computer Aided Verification,Berlin: Springer, 2009. 682-688.

[25] Lomuscio A, Qu H, RaimondiF. MCMAS: An Open-Source Model Checker for the Verification of Multi-Agent Systems.International Journal on Software Tools for Technology Transfer,2017, 19(1): 9–30.

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

[27] Wang F, Schewe S, Huang CH. An extension of ATL with strategy interaction. ACM Trans. on Programming Languages and Systems, 2015,37(3): 9:1-9:41.

[28] Huang CH,Schewe S, Wang F.Model-checking iterated games. Acta Informatica, 2017, 54(7): 625-654.