文章编号: 1000-6788(2012)06-1305-09 中图分类号: TP311 文献标志码: A

复杂自适应系统的 MAS 动态协作任务求解时序逻辑模型

蒋伟进 [1,2], 张莲梅 [1,3], 史德嘉 [1,2]

(1. 湖南商学院 计算机与电子工程学院, 长沙 410073; 2. 湘潭大学 信息工程学院, 湘潭 411000;
3. 武汉大学 电气工程学院, 武汉 430072)

摘 要 借鉴组织学思想将自适应系统中的自主运行单元抽象为 Agent, 把复杂自适应系统视为多 Agent 组织, 从时间和状态角度对复杂动态系统的行为进行描述, 提出了基于时序活动逻辑的多 Agent 动态协作任务求解自适应机制和构造模型; 分析了任务求解 BDI Agent 的信念、愿望、意图的产生过程和实现方法, 建立了协商推理的语义规则和行为规则, 给出了协作群组的选择算法. 并从任务求解 Agent 的心智变化角度, 描述了动态协作任务求解模型实现的六个阶段: 任务动态分配、协作意愿产生、协作群体生成、共同计划制定、协作群体行动和结果评估. 通过在 MAGE 等平台上实验和仿真测试, 验证了方法的可行性和有效性.

关键词 复杂自适应系统; 动态协作; 任务求解; 多 Agent 系统; 时序逻辑

The sequential logic model to solve the multi-Agent dynamic cooperative tasks of complex self-adaptive system

JIANG Wei-jin[1,2], ZHANG Lian-mei[1,3], SHI De-jia[1,2]

(1. School of Computer Science, Hunan University of Commerce, Changsha 410073, China;
2. School of Information Engineering, Xiangtan University, Xiangtan 411000, China;
3. Electrical Engineering College, Wuhan University, Wuhan 430072, China)

Abstract Solving dynamic complex problem is difficult in the theory and applied research of artificial intelligence and complex adaptive systems. Idea from histology is that the auto-run unit in self-adaptive system is abstracted to be Agent, the complex adaptive system is considered as a multi-Agent tissue. The behavior of complex dynamic systems in time and space is described. The adaptive mechanisms and structure model of solving multi-Agent dynamic cooperative tasks based on sequential active logic are proposed. The production process and realization of BDI belief, desire, intention of solving task are analyzed. The semantic rules and action rules of cooperative deduction is builded. The selection algorithm of cooperative groups is given. From the mind change of task solving Agent, the paper describes the six stages to realize the solving model of dynamic cooperative tasks. The six stages are dynamic allocation of tasks, collaboration will produce, generate collaborative groups, common planning making, collaborative groups action and evaluation of results. Experiments and simulation on MAGE and other platforms prove the feasibility and effectiveness of our proposed approaches.

Keywords complex self-adaptive system; dynamic cooperation; task solving; MAS; sequential logic

1 引言

近年来随着计算机技术和通信技术的迅猛发展, 出现了越来越多的基于 Web 和 WWW 技术的大型复杂系统. 这些复杂系统的特点是不仅问题所涉及的领域特别大, 内部关系十分复杂难以预测, 而且这些大型复杂问题的内部构成要素之间的关系是动态变化的. 这就要求系统的组织结构和问题求解模型能够适应环境的变化. 鉴于动态性这一新的特点出现, 使得原有的协作任务求解的一些理论和方法不再适合, 因此, 必须寻找和建立新的动态协作任务求解机制和方法.

收稿日期: 2010-04-28
资助项目: 教育部人文社科研究一般规划基金 (11YAZH039); 湖南省自然科学基金重点项目 (10JJ5064, 11JJ6051); 湖南省科技计划重点项目 (2009GK2002); 科技部中小企业技术创新基金 (09C26214301947)
作者简介: 蒋伟进, 博士, 教授, 研究方向: 数据挖掘, 多 Agent 系统, 协同计算, 社会计算.

目前, 已有许多学者采用多 Agent 系统 (multi-Agent system, MAS) 来构建协作任务求解系统, 已经开发出一些面向理论或面向实现的协作问题求解模型. 如, Hewitt 在提出分布式人工智能的开放信息系统 (open information system, OIS) 语义时指出, 需要用数学语言来建立 OIS 的社会学和并发系统科学的基础[1]. Werner[2] 建立的有关协同主体通信和社会结构的统一理论; Bond[3] 通过承诺建立的协同智能主体组织的计算模型; Brazier[4] 提出的用组合形式化框架模型化多主体系统; Wooldridge 和 Jennings[5] 基于 Jennings 提出的两个相对应的概念 – 承诺和约定而建立的描述整个协作问题求解过程的一个模型. Sandholm[6] 提出并实现了一个基于边际成本计算的 MAS 合同网协商协议. Rosenschein[7] 提出了一个基于权力与承诺的多 Agent 系统协作形式化模型. 李晶 [8] 结合描述 Agent 能力的 VSK-AF 逻辑 [9] 和描述 Agent 思维状态的协商推理, 构成了基于能力和思维的协商公理体系多 Agent 系统模型, 并引入 “能力”、“权力” 和 “授权” 等概念描述系统中 Agent 的动作选择策略. 蒋伟进等 [10] 研究了一种合作与竞争共存的 Agent 间关系, 建立了基于承诺与学习能力的 MAS 动态协作关系.

文献 [11] 将知识表述和推理系统的描述性结构 [12] 和任务逻辑的任务语义 [13] 结合起来, 构造了一个可以描述具体属性的、可判定的任务逻辑系统, 给出了系统的结构、语法、语义和推理规则. 从理论上讲, 用任务逻辑描述 Agent 之间的高层交互是个很好的选择. 但是, 这种任务逻辑系统存在一个缺点 —— 它是不可判定的 (仅是半可判定的), 以其为基础构造应用系统时无法保证推理过程都能在有限的时间内结束. 在现有的任务逻辑框架下, 可以用命题逻辑结构代替一阶谓词逻辑结构 [14], 使得任务逻辑可以判定, 但这样会使得任务逻辑系统的表达能力太弱, 无法满足应用系统的要求. 文献 [15] 针对这个问题, 在文献 [11] 的基础上给予了完善, 证明了任务的断定可完成性是可判定的, 同时说明了描述任务逻辑系统的可判定性、可靠性及完备性. 它克服了现有的任务逻辑的缺点, 可以提供较强的表达能力和可判定的推理服务, 在促进任务逻辑真正走向实用方面具有一定的意义, 也为其应用奠定了坚实的理论基础.

由于现有任务描述逻辑的任务的定义局限于抽象的、逻辑的定义, 没有将具体属性和任务联系起来, 尤其是动态复杂问题的出现, 使得任务求解的过程受到了时间的约束和限制. 因此, 为了更好的描述动态协作任务求解过程, 就必须使得任务求解表示逻辑能够体现出时间的特性. 本文提出用时序活动逻辑来描述系统的动态特征并建立动态协作任务求解过程. 该模型在感知处理、选择动作、确定动作阶段不考虑时间因素, 在协商和执行时才考虑时间, 并且假定时间经过处理后是线序、离散的, 在时序逻辑中增加时序算子. 用线性时序逻辑描述权力随时间的变化以及系统运行时, Agent 执行行为不受时间约束. 本文主要对基于时序逻辑的 BDI 多 Agent 系统协商模型以及相关推理规则和授权规则及其语义进行探讨.

2 基于时序活动逻辑的任务求解模型

2.1 TL 系统模型

基于时序活动逻辑 [15] 的动态协作 MAS 简称为 TL 系统, 它由时序结构 [16−18]、行为集合、环境、Agents 集合、协商过程和 Agents 财富组成.

定义 1 环境是一个元组 $Env =< E, S, W, Vis_1, Vis_2, \cdots, Vis_n, acce_1, acce_2, \cdots, acce_n, Arbitrage, \tau_g, e_o, s_o >$. 其中, $E = \{e_0, e_1, \cdots\}$ 是外部环境; $S = \{s_0, s_1, \cdots\}$ 是 Agents 环境; $W = \{w_1, w_2, \cdots\}$ 是任务集合; $Vis_i : E \to 2^E$ 是 $Agent_i$ 的外部环境状态划分函数; $acce_i : S \to 2^E$ 是 $Agent_i$ 的 Agents 环境状态划分函数; $Arbitrage : Message \times AS \to Message$ 是仲裁函数, 当协商无法解决冲突时用仲裁策略消解冲突; $\tau_g : E \times S \times Act(Arbitrage(msg)) \to 2^{E \times S}$ 是复合环境状态转移函数, 其中 $msg \in Message, Act : Message \to Actions^n$ 表示从动作选择状态消息中提取各个 Agent 的行为, $Actions^n$ 是 n 个 Actions 的笛卡儿积; $Act(Aribitrage(msg))$ 表示从经过仲裁的动作选择状态消息中提取的各个 Agent 的行为, 其中 Actions 是行为集合, $e_0 : e_0 \in E$ 是外部环境初始状态, $s_0 : s_0 \in S$ 是 Agents 环境初始状态.

定义 2 协商过程为 $N =< Ags, Isu, O, Vo, Ans, Time, Thread, Protocol >$. 其中, $Ags \subseteq \{Ag_1, Ag_2, \cdots, Ag_n\}$; Ag_i 为 $Agent_i$; $Isu = \{issue\}$ 是主题 issue 集合; O 为主题的范畴, 是一个知识体; V_o 为由 O 决定的主题的所有效取值的集合; $Ans = V_o \times Message$, 表示 Agent 在提议中提供的应答, 是关于协定的有效取值集合与动作选择状态消息集合的笛卡儿集. 其中关于协定的有效取值集合 $V_A = V_O \bigcup \{ACCEPT, REJECT, QUIT\}$, $ACCEPT$、$REJECT$ 和 $QUIT$ 分别表示接受、拒绝对方提议和退出协商; 动作选择状态消息集合 Message; Agent 通过在协商过程里使用 Message 体现在发生冲突时对自己行为的调整意见, 并把调整后

的消息发给其他 Agent; $Time = \{time'_1, time'_2, time'_3, \cdots, time'_n\}$; $Thread = \{Thread_{ijn}\}, Thread_{ijn}$ 表示 Ag_i 与 Ag_i 之间就主题 issue 的协商过程. 一个协商过程结束 (以 ACCEPT、QUIT、REJECT 结尾或者因协商时间长度到达规定而结束) 意味着某两个 Agent 的一次协商完成.

$$Thread_{ijn} = (On(isu_n, (ans(s_1, h_1, time_1)ans(s_2, h_2, time_2)\cdots))),$$

其中 $s_k, h_k \in \{i, j\}, time_k \in Time, ans(s_k, h_k, time_k) \in Ans$, 当 $k = 1(\mathrm{mod}2)$, 则 $s_k = i$ 和 $h_k = j$, 否则, $s_k = j$ 和 $h_k = i$, 是协商双方互发的提议中协定取值的交替序列;

Protocol: 用 KQML[7] 定义的协商协议.

定义 3 TL 系统是元组 $S =< Actions, NS, AS, Env, Agents, N, PoS, SR, T >$, 其中, $Actions$ 是动作集; NS 是协商策略集: AS 是仲裁策略集; Env 是环境; $Agents = \{Ag_1, Ag_2, \cdots, Ag_m\}$ 是 $Agents$ 集; N 是协商过程; $PoS \in Z$ 是系统财富; SR 是系统的社会规则; T 为时序结构. TL 系统的全局状态集合 $G = \{\varepsilon_0, \varepsilon_1, \cdots\}$ 是 $E \times S \times L_1 \times \cdots \times L_n$ 的子集合. 由所有 TN 系统组成的集合称为 TN 系统类, 记为 S.

定义 4 G 上的一个序列 $\varepsilon_0, \varepsilon_1, \cdots$ (是可枚举的) 表示系统 $S =< Actions, NS, AS, Env, Agents, N, PoS, SR, T >$ 的一次运行.

①初始状态

$$\varepsilon_0 = (e_0, s_0, \tau_1(l_1^0, see_1(Vis_1(e_0)), feel_1(acce_1(e_0))), \cdots, \tau_n(l_n^0, see_n(Vis_n(e_0)), feel_n(acce_n(e_0)));$$

②对所有的 u, 如果

$$\varepsilon_u = (e_u, s_u, l_{u1}, \cdots, l_{un}),\ \text{且}\ \varepsilon_{u+1} = (e_{u+1}, s_{u+1}, l_{nu+1}),$$

那么

$$(e_{u+1}, s_{u+1}) \in \tau_g(e_u, t_u, a_{uk1}, \cdots, a_{uki}),$$

且

$$l_{iu+1} = \tau_i(l_{ui}, see_i(Vis_i(e_u)), feel_i(acce_i(t_u))),$$

其中 $a_{uki} = SelectA\text{-}P_i(lu_i, Ability_i, Power_i)$, 它与其他 Agent 选择的行为不冲突或者通过协商决定由 Ag_i 执行 a_{uki}.

定义 5 给定一个 TL 系统 $S =< Actions, NS, AS, Env, Agents, N, PoS, SR, T >$, 如果 $\varepsilon \in G_s$, 当且仅当 ε 出现在 S 的运行中, 则称 $G_s \subseteq G$ 为由 S 产生的可达全局状态集.

该模型的运行模式为: 除正在执行行为的 Agents 外, 其它所有 Agents 观察环境是同步的; 并根据自己此时的局部状态、能力和权力选择一个动作, 这样, 就可能出现以下三种情况.

1) 某些 Agents 此时没有执行任何动作的能力和权力导致它什么也不能做, 等到下一时刻又重新观察环境、选择动作.

2) 选择动作的 Agents, 如果动作没有冲突则确定动作、执行动作. 与文献 [1] 中执行动作不同的是考虑时间, 完成某个动作可能需要在几个单位时间上, 环境状态发生转移.

3) Agents 所选择动作有冲突则进行协商, 其协商的结果出现 3 种情况: ①协商失败, 下一时刻重新观察环境, 选择动作. ②协商成功, 按达成的协定执行动作, 环境状态发生转移. ③协商还在进行, Agents 在下一时刻重新观察环境.

下面引入 Agent 协商描述语言 L_{TL} 表示 MAS 的 TL 系统的信息特征. 语言 L_{TL} 由两部分构成. 在感知处理部分使用多模态 VSK-AF 逻辑语言 $L_{VSK\text{-}AF}$[4], 它能够表达多 Agent 系统中的客观现象, 并且可以表示系统中 Agent 可访问的或可知道的信息以及感知到的信息, 还可表示 Agent 在系统中知道的信息; 另一部分, 在选择动作、协商和确定动作、执行动作部分的描述使用 L_{TN}, 它表示基于时序逻辑 Agent 执行任务的动作 $E_{xu}(Ag_i, a, w)$、能力 $Capability(Ag_i, a)$、权力 $Right(Ag_i, a)$, 以及 Agent 之间非限制授权 $NR - Entitle(Ag_j, Ag_i, a)$、Agent 之间授权的转移、剥夺 Agent 执行动作 a 的权力 $Deprive(Ag_i, Ag_j, a)$, 动作选择 $Select(Ag_i, a, w)$、协商过程 $Negotiation(Ag_{ts}, a, w)$, 在执行任务时行为发生冲突 $Collision(a, w)$.

在语言 L_{TL} 的复合公式中引入 5 个时序算子: □ (将永远), ◇ (将会有), ○ (下一状态), ◆ (过去有), U (直到).

定义 6 给定原子公式集 φ_0, 动作调用词元组集 II, L_{TL} 的复合公式 φ 定义如下:

① $True, False \in \varphi, \varphi_0 \subseteq \varphi$;

②若 $\varphi_1, \varphi_2 \in \varphi$, 则 $\neg\varphi_1 \in \varphi, \varphi_1 \bigwedge \varphi_2 \in \varphi, \bigcirc\varphi_1 \in \varphi, \blacklozenge\varphi_1 \in \varphi, \Box\varphi_1 \in \varphi, \Diamond\varphi_1 \in \varphi, U\varphi_2 \in \varphi$;

③若 $\varphi \in \varphi$ 且 $\pi \in II$, 则 $[\pi] \in \varphi, < \pi > \varphi \in \varphi$.

其中, $[\pi]\varphi$ 表示执行中规定的动作一定使 φ 为 True; $< \pi > \varphi$ 表示执行规定的动作可能使 φ 为 True.

2.2 面向任务求解 BDI Agent 结构

下面在时序活动逻辑的基础上, 提出基于活动的 BDI Agent.

图 1 为基于活动的任务求解 Agent 的结构. 它分别由任务分析、活动计划执行、执行结果评估、时序活动控制和执行状态校核及知识库六个部分构成. 其主要组成部分的具体作用为:

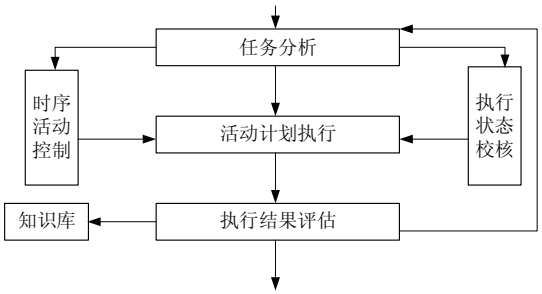

图 1 基于活动的任务求解 Agent 结构

任务分析: 通过对给定任务及环境 (包括内部环境及外部环境) 的分析, 产生可执行的活动计划及相应的时序活动逻辑关系表和计划执行的行为准则.

活动计划执行: 执行具体的活动计划, 实现任务的求解.

执行结果评估: 对执行结果与预期目标的一致性进行检测, 若为真, 则输出执行结果; 否则, 任务求解过程失败或反馈任务分析继续进行求解, 直到获得与预期目标相一致的结果.

知识库: 对任务求解过程中积累的经验进行存储的部件, 体现了主体的学习能力.

2.2.1 任务分析

任务分析包括信念、愿望及意图的形成过程. 为述说方便, 下面先给出动态协作任务求解过程中的活动定义, 并在此基础上, 描述本研究中信念、愿望及意图的形成过程 [19−22].

1) 动态协作任务求解中的活动

对于动态协作任务求解而言, 任务求解的全过程称为活动. 一个活动由若干个行为序列组成.

定义 7 对于任务 T 而言, 它的执行过程可以称为一个活动 h, $h = \{\alpha_1, \alpha_2, \cdots, \alpha_n\}, \forall\alpha_i \in h$, 则称 α_i 为活动 h 的一个动作, 任一活动 h 由若干个动作序列组成.

对于任意一个活动而言, 它可以由若干个动作序列, 这些动作序列具有一定的次序, 求解实体按次序执行相应的动作, 从而完成整个任务的求解.

对于任意活动 $h_i(1 \le i \le n)$ 也可以分为若干个子活动, 若不可分, 则称其为一个基本 (原子) 活动; 同理对 $\forall\alpha_i \in h$, 也可以分解为若干个子序列, 若不可分, 则称其为一个基本 (原子) 动作.

2) 信念形成

Agent 的信念是指 Agent (或者是关于自身) 的信息, 通过这些信息所形成的对环境 (或自身) 的认识称为信念. 对于面向任务求解的实体而言, 它的信念不仅包括对环境和自身的认识, 而且还包括对任务的认识, 这种认识表现为形成可能实现任务求解目标的所有可能的活动集. 具体形成过程为:

i) 对任务 T 的可行性进行分析, 若可求解该任务, 则创建基于 Agent a 的任务求解的预期实现目标 g_{pre}.

ii) $Analyse(T) \to character_T \bigwedge Observe(E) \to knowledge_E$.

iii) $C\_Analyse(character_T, knowledge_E) \to H_p$.

定义 8 对于活动集合 $H_p = \{h_1, h_2, \cdots, h_n\}, \exists\forall h_i \in H_p(1 \le i \le n)$ 都有 Agent $Guess(h_i, g_{pre})$, 则称活动集合 H_p 为给 Agent 的信念集合 belief. $Guess(h_i, g_{pre})$ 表示 Agent 的一种心智状态, 即 Agent 推测通过活动 h_i 可能实现预定目标 g_{pre}.

对于 $\exists\forall h_i \in H_p(1 \le i \le n)$ 不一定能够实现预定目标 g_{pre}, 可能不能实现预定目标 g_{pre}, 也可能是满足一定条件下才能够得以实现.

3) 愿望形成

愿望是指可以实现的信念集合, 而对于动态协作任务求解而言, 愿望是指在当前环境下有可能实现的意图. 环境是动态变化的, 因此需要对原有关于环境的认识进行修正, 具体过程为:

i) $Modify(knowledge_E, knowledge_E) \to knowledge_{E_a}$.

ii) $Filter(Belief, knowledge_{E_n}) \to H'_p$.

定义 9 对于活动集合 $H'_p = \{h'_1, h'_2, \cdots, h'_n\}, \exists \forall h'_i \in H'_p (1 \le i \le n)$, 都有 Agent $Believe(h_i, g_{pre})$, 则称活动集合 H_p 为该 Agent 的信念集合 $Desire$. 同理, $Believe(h_i, g_{pre})$ 表示 Agent 相信在当前环境下通过活动 h_i 能够实现预定目标 g_{pre}.

对于组成愿望集合中的 $\exists \forall h'_i \in H'_p (1 \le i \le n)$, 一定是在当前环境下能够实现预定目标 g_{pre} 的活动, 但是任意两个活动之间可能上相互矛盾的.

4) 意图形成

意图是指可实现的愿望. 在动态协作任务求解中, 从愿望集合中选择出最优的活动所形成的集合就是意图. 因此, Agent 所找出的能够满足的愿望子集, 就构成了目标集. 其具体形成过程为:

i) $Analyse(a) \rightarrow character_a$. 对任务执行体 a 进行分析 $Analyse(a)$, 形成执行体的特征集合 $character_a$.

ii) $Filter^*(desire, character_a) \rightarrow H^*_p$. 根据执行体的特征 $character_a$, 对愿望集合 $desire$ 进行优选 $filter^*(desire, character_a)$, 选择出最适合该任务执行体执行的活构成集合 H^*_p, H^*_p 中的可能存在多个活动, 也可能存在唯一的活动.

定义 10 对于活动集合 $H^*_p = \{h^*_{p1}, h^*_{p2}, \cdots, h^*_{pn}\}, \exists \forall h^*_{pi} \in H^*_p (1 \le i \le n)$, 都有 $Agent\, Confirm(h_i, g_{pre})$, 则称活动集合 H_p 为该 Agent 的集合 $desire$. $Confirm(h_i, g_{pre})$ 表示 Agent 确信通过活动 h_i 一定能够实现预定目标 g_{pre}.

对于组成愿望集合中的 $\exists \forall h'_i \in H'_p (1 \le i \le n)$, 一定是在当前环境下最符合该任务执行体特点的活动.

2.2.2 任务执行

在 Agent 所形成的意图集合中, 各个活动集合是孤立存在的, 并不是一个有机的整体, 而且很有可能各个活动集合之间是存在着矛盾. 为了能够保证 Agent 顺利实现预定目标 g_{pre}, 就必须对意图集合中的活动按时间顺序组成相应的时序活动序列, 这个时序活动序列就是 Agent 的活动计划 p_a. 对于每一个活动 $\exists \forall h^*_{pi} \in H^*_p (1 \le i \le n)$ 规定相应的时序, 就形成了稳态活动 β_i. 由于活动的执行方式有两种, 活动计划 p_a 就是由若干个稳态活动按串行关系 $\downarrow$ 与并行关系 $\uparrow$ 构成的时序活动序列.

定义 11 对于时序活动序列集合有行动计划

$$P_{action} = \{\beta_{p1}, \beta_{p2}, \cdots, \beta_{pn} \mid \forall \beta_{pi}, \beta_{pj}, \exists \beta_{pi} \downarrow \beta_{pj} \vee \beta_{pi} \uparrow \beta_{pj}, 1 \le i, j \le n\},$$

其中, $\beta_{a_i} = (< t_{ph}, t_{pi} >, h^*_{pi}), \beta_{a_j} = (< t_{pi}, t_{pj} >, h^*_{pj}), \exists \forall h^*_{pi}, h^*_{pj} \in H^*_p (1 \le i, j \le n)$.

该行动计划制定后, 任务求解 Agent 就按所制定的行动计划执行. 若任务求解 Agent 所处的环境是静态的, 则按所制定的行动计划就能够实现预期目标 g_{pre}. 但是, 为了更好地保证任务执行过程顺利实现, 对组成计划的每个稳态活动必须执行具体的实现目标和评估条件 (即行为准则), 并依据标准进行验证保证行动计划的顺利实现. 另一方面, 由于目前的任务都是动态复杂任务, 标准集合是随时间变化而不断变化的, 这样校核的过程也会随之变化.

定义 12 活动准则 $R = \{(g_{\beta_i}(t), C_{\beta_i}(t)) \mid i \in (1, 2, \cdots, n)\}$, 其中 g_{β_i} 为行为计划中 β_i 的具体实现目标, c_{β_i} 为相应的评估条件, $g_{\beta_i}(t)$ 与 $C_{\beta_i}(t)$ 分别表示它们随时间而变化.

依据具体的活动准则行为 r_i 对任务求解 Agent 活动计划进行校验 $verify(\beta_i, r_i)$. 若 $?verify(\beta_i, r_i) = true$, 则实现具体的实现目标, 按活动计划继续执行. 若 $?verify(\beta_i, r_i) = false$, 则调整相应的稳态活动 $adjust(\beta_i, r_i) \rightarrow \beta'_i$ 产生新的稳态活动 β'_i. 调整行为相当复杂, 有时可能影响整个计划, 这里不做深入分析, 我们将另文专题介绍.

2.2.3 结果评估

任务求解 Agent 执行活动计划, 得到任务求解的结果. 理想状态下, 由于活动计划 P_c 的制定与预期目标 g_{pre} 是相一致的, 因此任务求解结果 S_T 应能满足或接近预期目标 g_{pre}. 但是, 由于任务求解的过程是动态变化的, 整个求解过程存在着很多不确定性, 因此需要对所产生的结果进行评估, 而且通过评估可以找出求得结果与预期目标之间存在的差距. 结果评估就是对求得的结果进行一致性校验 $accord(S_T, g_{pre})$. 若 $?accord(S_T, g_{pre}) = true$, 则该结果符合预期目标 g_{pre}; 否则为不符合, 那么任务求解过程失败或新的求解过程开始.

3 动态协作任务求解

3.1 共同目标

当任务求解 Agent 产生了协作意愿之后 (我们称其为协作发起者), 则他通过通信的方式 [23−24], 向其他 Agent 发出协作意愿请求即 $require(a', Will(a, g_{pre}))$, 其中 a' 表示 Agent 集合中的某个特定求解 Agent 或某些求解 Agent. 其他求解 Agent 收到它的协作请求后, 根据自身的情况做出 $accept(Will(a, g_{pre}))$ (接受请求)、$reject(Will(a, g_{pre}))$ (拒绝请求) 或不做回应三种不同的反应.

当发出请求的求解 Agent 收到来自某其他 Agent 的接受请求应答 (我们称这种 Agent 为接受者) 后, 就通过通信机制将自己的预期实现目标 g_{pre} 传送给接受者.

接受者收到发起者的 g_{pre} 后, 进行一致性检查 $?consistency(g_a, g'_a)$. 若为 true, 则说明二者的预期实现目标一致 (即共同目标, 记为 g_{common}). 接受者向发起者发出同意协作即 $agree(Will(a, g_{pre}))$, 即此时二者的协作意愿变成现实. 当然, 在动态协作任务求解过程中, 存在多个求解 Agent, 可能存在一个接受者接受来自多个发起者的请求或它同时向其他 Agent 也发出协作请求的复杂情况.

3.2 任务 Agent 群组的选择算法

定义 13 任务 Agent 选择算法用五元组表示: $< B_i, P_j, S_o, C_o, G >$[25−26]. 其中 B_i 是群组中的用户 Agent 集合, P_j 是从 B_i 中选择的任务 Agent 集合, S_o 是从 B_i 中选取的待确定是否成为任务 Agent 的用户 Agent,

C_o 为协同 Agent, 而 G 则为待解的任务 [27−28]. 同时定义如下一些元操作 (primitives)[29−30]:

$Request(C_o, S_o, G, [date])$: 表示 C_o 要求 S_o 加入任务群组, 一起完成任务 G 的执行. $[date]$ 是可选项, 若有此参数表明 C_o 给 S_o 回复的限制时间. 根据实际需要也可以表达其它的意思或再加入其它的参数;

$Reject(S_o, C_o, G)$: 表示 S_o 拒绝 C_o 的协同请求;

$Wait\_notice(S_o, C_o, G)$: 表示 S_o 通知 C_o 正在决定是否参与 G 的协同执行;

$Accept(S_o, C_o, G)$: S_o 通知 C_o 它同意参与协同任务的执行;

$Accept\_affirm(C_o, S_o, G)$: C_o 通知 S_o 它成功加入任务群组, 参与协同任务的执行;

$Add\_P(S_o, P_j)$: S_o 被加入到 P_j 中, 成为任务群组的一个成员;

$Quit\_request(S_o, C_o, G)$: S_o 向 C_o 发出退出任务群组请求;

$Quit\_accept(C_o, S_o, G)$: C_o 接受 S_o 退出的请求;

$Callback(C_o, S_o, G)$: C_o 回收分配给 S_o 的子任务及中间结果与状态;

算法 1 群组选择算法

输入初始信息: 用户 Agent 集合 B_i, 待解决的任务 G 和初始条件.

输出结果信息: 选择的任务 Agent 集合 P_j.

Step 1 如果位于 B_i 中的任务发起者用户 Agent a_i 能独立完成任务, 则转 Step 15;

Step 2 a_i 发一个任务协同解决的请求给协同 Agent C_o;

Step 3 $Add\_P(a_i, P_i)$;

Step 4 C_o 根据 FSIB 从 B_i 中选择一个还没有被选择过的用户 Agent 作为 S_o;

Step 5 如果找不到合适的 S_o, 并且 P_i 不能完成任务, 则任务不可解, 转 Step 16;

Step 6 $Request(C_o, S_o, G, [date])$; C_o 向 S_o 发出协同请求;

Step 7 $Wait\_notice(S_o, C_o, G)$; 在给定的时间内若不能及时响应 C_o 的请求则给予一个等待回复;

Step 8 如果等待的时间超过 $Request(C_o, S_o, G, [date])$ 中 date 的限制, 则发出 $Reject(S_o, C_o, G)$, 转 Step 4;

Step 9 如果接收到消息 $Wait\_notice(S_o, C_o, G)$, 则转 Step 6;

Step 10 如果接收到消息 $Accept(S_o, C_o, G)$, 则做 $Accept\_affirm(C_o, S_o, G), Add\_P(S_o, P_i)$;

Step 11 如果 G 还不能由 P_j 完成, 则转 Step 4;

Step 12 C_o 根据任务分解算法将任务进行分解并调用分配算法分配子任务给 P_j;

Step 13 各任务 Agent 并发执行分配的子任务;

Step 14 如果有任务 Agent 执行 $Quit\_request(S_o, C_o, G)$ 并且分配给他的子任务没解决完毕, 则 C_o 做出响应, 若 $Quit\_accept(C_o, S_o, G)$, 则调用 $Callback(C_o, S_o, G)$ 回收其任务状态 (包括子任务中间结果) 返回

到 Step 4, 否则作异常处理;

Step 15 C_o 从各个任务 Agent 获得执行的结果以合并成最终的结果;

Step 16 算法结束.

4 仿真及结果分析

为了对本文的任务协作逻辑模型进行有效性验证, 我们在 MAGE 平台上进行了仿真实验. MAGE 平台 [31] 是中国科学院智能信息处理重点实验室开发的一种面向主体的软件开发、集成和运行环境, 其自带了一系列的工具支持面向主体的软件开发和系统集成. MAGE 平台提供了面向 Agent 的软件开发模式, 以 Agent 为基本的构件单元, 从而可以方便、高效地构建大型应用系统.

测试运行环境为: 4 台双核处理器机器, Processor Intel Pentium D CPU 3.00GHz; RAM 2GB total; OS GNU/Linux (kernel release 2.6); Compiler Java 2 Standard Edition v.1.6 with 1024MB Java VM memory; MAS MAGE.

首先, 我们构造了一些公式, 比如 $\varphi :=< \pi_1 \quad \pi_2 > (\neg < C_1(x)? > (C_2 \quad C_3)(x))$. 当检测此类公式 φ 的可满足性, 应用扩展规则对初始划分进行扩展时, 将生成相互独立的 6 个划分. 并且保证除此以外没有其它的划分, 于是在本系统中同时存在 6 个推理子任务.

图 2 表示逐步加速的效果.

为了在一个更真实的环境下评估系统的推理性能, 我们建立了一个本体, 该本体包含 47 个概念, 50 个角色, 17 个原子动作, 22362 个个体, 216659 个三元组. 我们使用 MAGE 计算在该本体上的一些常见的推理问题, 例如: 包含性问题 (subsumption)、概念可满足性问题 (concept satisfiability)、个体从属判断 (instance checking). 最后统计协商推理的性能, 如图 3 所示.

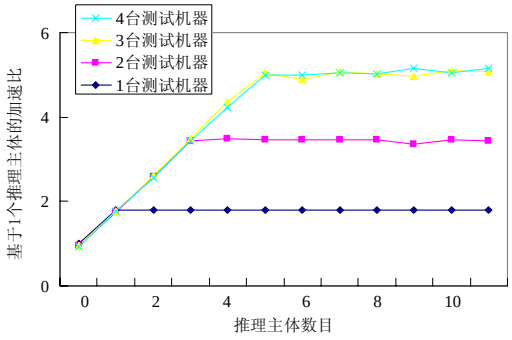

图 2　TLMAS 协作推理加速比

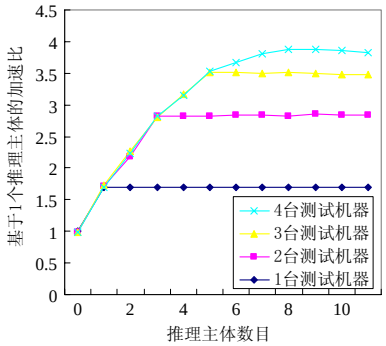

图 3　TLMAS 协作推理加速比

可以观察到在两个测试实验中都有一个明显的推理加速. 在第一个测试中, 最多存在 6 个可以并行计算的推理任务, 因此随着推理 Agent 数目的增加, 加速比不可能一直增加. 而在第二个测试中, 当使用 4 台机器进行测试时, 随着推理 Agent 的数目增加, 推理性能略有下降可能有两方面的原因. 首先, 在测试用例 2 中, 频繁应用不确定的扩展规则, 从而产生大量的分支划分. 这将导致大量的内存分配操作, 当系统负荷过重的时候, 推理 Agent 会迁移到局域网上的其他机器上, 这又增加了网络开销. 其次, 产生大量的分支, 也会导致任务池的同步开销大大增加. 这个额外的开销正比于并行执行的推理 Agent 数目.

此外, 为了更进一步验证和应用本文模型的有效性, 我们还在面向 P2P 社区的任务协作仿真中, 将本文的模型与传统合同网 [32](CNM) 模型和文献 [33] 中的动态合同网 (DCNM) 进行了实验比较. 下面首先给出一些实验参数的定义.

恶意节点 (Agent) 参与率: 恶意节点参与率就是在协作任务中参与任务的是恶意节点的比率. $\alpha = htag/tal$, 其中, α 为恶意节点参与率, $htag$ 为任务协作中的恶意节点, tal 为参与协作任务都总 Agent 数 (节点).

任务完成效率: $\beta = SucAmount/TotalAmount$ 其中, β 为任务完成效率, $SucAmount$ 为完成的子任务数, $TotalAmount$ 协作任务的所有子任务数.

在本文任务协作逻辑模型中, 我们通过一系列的规则来提高模型在抵制恶意节点以及提高任务完成效率方面的性能, 并将本文的模型 TCLM-P2P 与传统合同网 (CNM) 和动态合同网 (DCNM) 进行比较. 设计了

一个多线程 JAVA 程序, 用来模拟同种环境下各个模型在性能方面的表现情况. 其中线程用来模拟各个模型, 每一个类型的线程都附有计算功能. 实验模拟结果如图 4 和图 5 所示.

图 4 是在善意节点 300 个, 恶意节点和预先信任节点各 40 个的情况下三个模型抵制恶意节点方面的表现.

由图 4 可以看出, 本文模型在整体性能上要优于 CNM、DCNM, 并且在稳定性上也要优于前两种模型, 这主要是因为在 CNM 和 DCNM 中节点被假设为是完全可信的, 并没有相应的规则来抵制恶意节点, 导致了协作任务中恶意节点的参与率较高. 而我们的 TCLM-P2P 中, 提出了一系列的激励规则来提高可信节点的协作任务参与率, 降低恶意节点对协作任务的影响.

图 5 表示的是三个模型在任务完成率方面的比较结果. 实验假定在某一 P2P 网络社区中一次协作任务包含 100 个子任务.

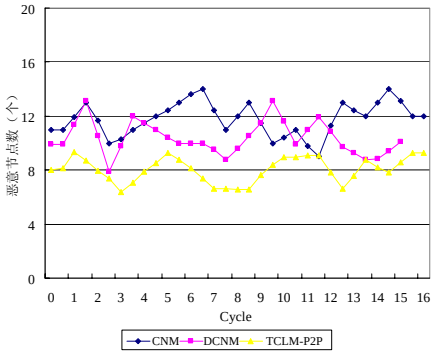

图 4 三个模型抵制恶意节点的比较

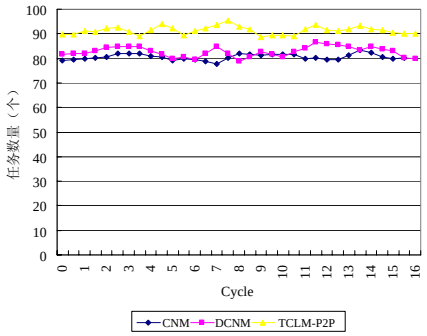

图 5 任务完成效率的比较

通过图 5, 可以发现在任务完成效率上, 本文的模型要优于 CNM 和 DCNM. 这主要是因为本文的模型通过协作任务奖励实现了激励规则, 提高了节点参与协作任务的积极性, 并且按照一定的规则来选择协作节点, 从而达到提高任务完成率的目的.

5 结论与展望

本文借助于组织学思想将自适应系统中的自主运行单元抽象为 Agent, 把复杂自适应系统视为多 Agent 组织, 从时间和状态角度对复杂动态系统的行为进行描述, 提出了基于时序活动逻辑的多 Agent 动态协作任务求解自适应机制和构造模型及其实现方法. 进一步需要解决的问题是在进行目标及任务划分时, 如何能够达到合理分割, 从而使得在进行协作的过程中, MAS 之间的通信量或是协作的复杂度降低, 从而提高整体的工作性能. 同时, 在开放的、动态的复杂环境中, 希望通过更多的、系统所处环境更为复杂的案例的分析和研究来验证本文工作的有效性.

参考文献

[1] Hewitt C. Open information systems semantics for distributed artificial intelligence[J]. Artificial Intelligence, 1991, 47: 79–106.

[2] Werner E. Cooperating Agents: A Unified Theory of Communication and Social Structure[M]// Gasser L, Huhns M N. Distributed Artificial Intelligence, London: Pitman / Morgan, 1989: 3–36.

[3] Bond A H. Commitment: A computation model for organization of cooperating intelligence agent[C]// Proceeding 1990 Conference on Official Information System, Cambridge, MA, 1990.

[4] Brazier F M T, Dunin-Keplicz B M, et al. DESIRE: Modeling multi-agent systems in a compositional formal framework[J]. Journal of Cooperative Information Systems, 1997, 6(1): 67–94.

[5] Wooldridge M, Jennings N R. Towards a theory of cooperative problem solving[R]. N. R. Jennings@qmw.ac.uk.

[6] Sandholm T. An implementation of the contract net protocol based on marginal cost calculations[C]// Communication in Multi-Agent Systems: Agent Communication Languages and Conversation Policies, Melbourne, Australia, 2003: 51–97.

[7] Rosenschein J S, Zlotkin G. Rules of Encounter: Designing Conventions for Automated Negotiation Among Computers[M]. Boston: MIT Press, MA, 1994.

[8] 李晶. 基于能力与权力的协商公理体系多 Agent 系统模型 [J]. 云南师范大学报, 2003, 23(2): 23–26.
Li J. Multi-agent system model based on the abilities and powers of negotiation axiom system[J]. Journal of Yunnan Normal University, 2003, 23(2): 23–26.
[9] 孙瑜. 多 Agent 系统模型及其应用 [J]. 计算机工程与应用, 2003, 39(12): 50–53.
Sun Y. Multi-agent system model and its application[J]. Computer Engineering and Applications, 2003, 39(12): 50–53.
[10] Jiang W J, Xu Y S. Commitment and learning capabilities based on dynamic collaborative relationships MAS[C]// Proceedings of 9th Pacific Rim International Workshop on Multi-Agents, PRIMA Agent Computing and Multi-Agent System: Springer Verlag, Heidelberg, D-69121, Germany. LNAI, 2006, 4088: 608–613.
[11] Japaridze G. The logic of tasks[J]. Annals of Pure and Applied Logic, 2002, 117(1/3): 263–295.
[12] Baader F, Sattler U. An overview of tableau algorithms for description logics[J]. Studia Logica, 2001, 69: 5–40.
[13] Baader F, et al. The Description Logic Hanndbook: Theory, Implementation and Applications[M]. Cambridge: Cambridge University Press, 2002.
[14] Baader F, Hanschke P. A scheme for integrating concrete domains into concept languages[R]. Deutsches Forschungszentrum Fur Kunstliche Intelligenz, Kaiserslautern: Research Report RR-91-10, l991.
[15] Japaridze G. Introduction to computability logic[J]. Annals of Pure and Applied Logic, 2003, 123(1/3): 1–99.
[16] 张会, 李思昆. 描述任务逻辑及其应用 [J]. 计算机学报, 2006, 29(3): 488–494.
Zhang H, Li S K. Describe the task logic and its applications[J]. Journal of Computers, 2006, 29(3): 488–494.
[17] Blass A. A game semantics for linear logic[J]. Annals of Pure and Applied Logic, 1992, 56(1/3): 182–220.
[18] Japaridze G. A constructive game semantics of the language of linear logic[J]. Annals of Pure and Applied Logic, 1997, 85(2): 87–156.
[19] 蒋伟进, 张莲梅, 王璞. 基于 MAS 协作机制的动态计算资源优化调度模型 [J]. 中国科学 F, 2009, 39(9): 977–989.
Jiang W J, Zhang L M, Wang P. Research on grid resource scheduling algorithm based on MAS cooperative bidding game[J]. Chinese Science F, 2009, 39(9): 977–989.
[20] Jiang W J, Liu D R. Research on agent motion-planning algorithm based on biology regulation mechanism[J]. Journal of Computational Information Systems, 2007: 1801–1808.
[21] Shi D J, Zhang L M. A novel research on multi-agent cooperation mechanism[J]. Journal of Information and Computational Science, 2008, 12: 2049–2055.
[22] Jiang W J, Zhang L M, Shi D J, et al. Research on distributed solution strategy of complex system based on MAS[J]. Journal of Information and Computational Science, 2008, 10(5): 2063–2069.
[23] 蒋伟进, 王璞. 基于 MAS 的复杂系统分布式求解策略与推理研究 [J]. 计算机研究与发展, 2006, 43(9): 1615–1623.
Jiang W J, Wang P. Solving strategies and reasoning based on the complexity of the MAS system distributed[J]. Computer Research and Development, 2006, 43(9): 1615–1623.
[24] 蒋伟进, 王璞. 基于 MAS 市场机制的动态计算资源调度模型研究 [J]. 计算机研究与发展, 2007, 44(1): 29–36.
Jiang W J, Wang P. Computing resources MAS market mechanism-based dynamic scheduling model[J]. Computer Research and Development, 2007, 44(1): 29–36.
[25] 盛秋戬, 赵志崑, 刘少辉, 等. 多主体团队交互协议 [J]. 软件学报, 2004, 15(5): 689–696.
Sheng Q J, Zhao Z K, Liu S H, et al. A teamwork protocol for multi-agent[J]. Journal of Software, 2004, 15(5): 689–696.
[26] Shi Z Z, Dong M K, Jiang Y C, et al. A logical foundation for the semantic web[J]. Science in China, Ser F, 2005, 48(2): 161–178.
[27] 常亮, 史忠植, 邱莉榕, 等. 动态描述逻辑的 Tableau 判定算法 [J]. 计算机学报, 2008, 31(6): 1–14.
Chang L, Shi Z Z, Qiu L R, et al. Dynamic description logic the Tableau determine algorithm[J]. Journal of Computers, 2008, 31(6): 1–14.
[28] 蒋运, 史忠植, 汤庸, 等. 一种分布式动态描述逻辑 [J]. 计算机研究与发展, 2006, 43(9): 1603–1608.
Jiang Y, Shi Z Z, Tang Y, et al. A distributed dynamic description logic[J]. Computer Research and Development, 2006, 43(9): 1603–1608.
[29] 董洁, 尹怡欣, 彭开香. 流程工业多智能体系统协调控制 [J]. 系统工程理论与实践, 2008, 28(10): 119–124.
Dong J, Yin Y X, Peng K X. An industrial process coordinated control based on multi-agent technology[J]. Systems Engineering — Theory & Practice, 2008, 28(10): 119–124.
[30] Horrocks I. DAML+OIL: A description logic for the semantic web[J]. Bull of the IEEE Computer Society Technical Committee on Data Engineering, 2002, 25(1): 4–9.
[31] Shi Z Z, Zhang H J, Dong M K, et al. MAGE: Multi-agent environment[C]// 2nd International Conference on Computer Networks and Mobile Computing, Shanghai, China: IEEE Computer Soc, 2003: 181–188.
[32] Zhang H J, Shi Z Z. Dynamic contract net protocol[J]. Computing Engineering, 2004, 30(21): 44–46.
[33] Smith R G. The contract net protocol: High-level communication and control in a distributed problem solver[J]. IEEE Transaction on Computer, 2005, C-29(12): 1104–1113.