CRQA_{OVTM} Agent 支持的开放环境下协同制造装配

周尤明 [1] 古华茂 [2]

摘 要 针对开放式环境下松耦合程度和可靠性不够的协作模式的缺点, 以及协同制造装配问题出现的 "结构失配" 和 "工艺革新" 的特性, 提出 CRQA_{OVTM} Agent 模型. 模型从服务参数 (属性) 松耦合程度和服务执行结果确定性两个方面区别定义普通 $(O, Ordinary)$ 接口、转换 $(V, conVersion)$ 接口、多参数适配器 $(T, multiparameteradapTor)$ 接口和概念实例调制器 $(M, conceptinstanceModulator)$ 接口, 并在此基础上提供包括普通协作、转换性协作、适配性协作和调制性协作的混合协作模式, 且在处方的基础上引入量化合成, 以更加灵活的方式处理 "结构失配" 和 "工艺革新" 问题. 同时模型引入合同的约束, 保证自组织协作是可信的. 通过验证并与 RPA_{CTI} Agent 模型比较可知, 该模型的协作机制既具有很高的成功率和效率, 又具有很高的灵活性, 更加适合开放环境下的协同制造装配.

关键词 CRQA_{OVTM} Agent, 协同制造装配, 合同约束, 多 Agent 系统

引用格式 周尤明, 古华茂. CRQA_{OVTM} Agent 支持的开放环境下协同制造装配. 自动化学报, 2018, **44**(7): 1333−1344

DOI 10.16383/j.aas.2017.c170048

Cooperative Manufacturing and Assembling Supported by CRQA_{OVTM} Agent in Open Environment

ZHOU You-Ming[1] GU Hua-Mao[2]

Abstract For the shortcomings of the cooperation mode of low loose coupling degree and low reliability in open environment, and for the "mismatched-structured" and "technics-innovation" characteristics of cooperative manufacturing and assembling problem, a CRQA_{OVTM} Agent model is presented. Four cooperative interfaces, called Ordinary, conVersion, adapTor and Modulator interfaces, are defined and distinguished by their different degrees of loose coupling of service parameters (attributes) and determinations of service execution result. Further based on them, a hybrid collaboration mode is presented, including Ordinary, conVersion, adapTor, and weakly-constrained Modulator collaborations. Based on recipes, quantitative service compositions are imported. All these make agents have the ability to cope with the "mismatched-structured" and "technics -innovation" problems in a more flexible manner. Contract constraint is also integrated into the model to ensure the self-organizing collaboration is trusted. Through verification and comparison with RPA_{CTI} Agent model we know, it strongly supports multi-agent systems to work well for cooperative manufacturing and assembling in open environment, by the guarantee of high success rate, high efficiency and more flexibility at the same time.

Key words CRQA_{OVTM} agent, cooperative manufacturing and assembling, contract constraint, multi-agent system (MAS)

Citation Zhou You-Ming, Gu Hua-Mao. Cooperative manufacturing and assembling supported by CRQA_{OVTM} agent in open environment. *Acta Automatica Sinica*, 2018, **44**(7): 1333−1344

鉴于 Agent 软件代理所特有的社会性、主动性、智能性和自治性, 基于多 Agent 系统 (Multi-agent system, MAS) 的分布式协同制造装配成为协同制造装配领域的一种有效方式[1−2]. 协同制造装配系统的一般抽象 MAS 模型可总结为 $\mathrm{MaAsS}=(Task, IniAg, Plan, PartAg, ServSet, EffiImp)$. 其中, MaAsS (Manufacturing and assembling system) 是制造装配系统, $Task$ 是制造装配任务, 包括制造和装配. 制造是指地理位置可以分散的原材料、零部件、半成品经过打磨、设计、组装等制造工序生产出成品的过程, 装配是指将成品所需

收稿日期 2017-01-19 录用日期 2017-05-31
Manuscript received January 19, 2017; accepted May 31, 2017
浙江省教育厅项目 (Y201533910), 浙江省自然科学基金 (LY15F020005), 全国教育信息技术研究课题 (156232397), 市级科技特派员项目 (2015KTZ21) 资助
Supported by Project of Zhejiang Education Department of China (Y201533910), Natural Science Foundation of Zhejiang Province (LY15F020005), National Education and Information Research Project of China (156232397), and Science and Technology Commissioner Project of City Huzhou (2015KTZ21)
本文责任编委 赵千川
Recommended by Associate Editor ZHAO Qian-Chuan
1. 湖州广播电视大学物流与信息工程学院 湖州 313000 2. 浙江工商大学计算机与信息工程学院 杭州 310018
1. College of Logistic and Information Engineering, Huzhou Radio and Television University, Huzhou 313000 2. College of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018

的零部件组装成成品的生产, 是制造的最后一个生产阶段. $IniAg$ 是接受 $Task$ 的发起者 Agent, 一般负责任务分解和分配, $Plan$ 是 $IniAg$ 任务分解和分配的规划, $PartAg$ 是参与任务求解的代理集合, $ServSet$ 为代理提供的服务 (包括资源) 集合, $EffiImp$ 是协作性能改进方案, 例如通过招标、拍卖、合同网等形式更好地选择代理完成任务或者更好进行资源配置. 制造装配问题一般首先要通过对任务进行规划[3−4] 预估, 事先得出制造装配的方案 (规划) 或者协同制造装配的具体 MAS 模型, 然后按照预先设计好的方案或者 MAS 模型进行制造装配. 一般说来面向协同制造装配问题的多 Agent 解决方法主要有两种: 第一种方法采用紧耦合的协作方式, 即是按照设计好的解决方案 (规划) 进行, 或者为方案的实施搭建体系结构. 在此过程中每个 Agent 都是特定工艺参数的 Agent, 不同 Agent 的协作工艺参数是固定的, 并且协作流程也是集中控制的, 不能以松耦合的方式管理 Agent 的协作行为[5−7]; 另一种方法把协同制造装配问题看成一般模型实例化的 MAS 问题[8−11], 按照具体 MAS 模型进行协同制造装配, 一般偏重于性能改进的方案研究, 没有改变任务结构本身以及协作的紧耦合本质. 不难发现, 这两种模型都面临实际制造装配中出现的 "结构失配" 与 "工艺革新" 的独有特点, 即是预先得出的规划只是制造装配过程的大致纲要, 制造装配实际过程往往与事先的规划存在偏差 (结构失配), 进而制造装配不能完全按照规划好的方案或者模型进行, 需要革新地调整规划、转换协作参数 (工艺革新), 而且只知道革新后的效果, 怎样革新工艺事先不知道.

有学者提出松耦合的 RPA_{CTI} Agent 模型[12], 但是也仅仅从服务应用域进行松耦合处理, 未能从服务参数上进行松耦合处理, 并且完成同一任务的可选路径 (方案) 少, 导致提供服务的接口松耦合度低, 可靠性也差. 以上方法都没有很好应对开放环境下协同制造装配的上述新问题.

为了有效解决协同制造装配上述新问题, 文中的 CRQA_{OVTM} Agent 模型对传统的 Agent 模型进行了扩展. 模型概括如下: 1) 在供需协作关系上, 考虑到服务请求与服务提供存在参数类型不配的问题 (工艺需革新问题), 扩展了传统的协作模式, 以普通接口、转换接口 ($conversion$ 缩写 V)、多参数适配器接口和概念实例调制器协作接口作为协作接口描述, 可以有效提高松耦合度 (提高灵活性), 并使接口具备工艺革新的特点, 多种接口的混合使用可以有效调整规划, 且在接口定义时增加可选路径, 提高协作的可靠性; 2) 从分布式业务流程上调整规划, 即是在基于处方的确定性规划基础上, 引入量化合成, 用于按需调整规划; 3) 由于存在不确定性协作, 通过协作关系的自修复、自演化来实现系统的稳定; 4) 通过合同表达 Agent 协同约束, 从而使得制造装配问题协同可信. CRQA_{OVTM} Agent 模型引入了多种协作接口, 同时增加量化合成, 引入松散性的自组织协作连接模式, 引入非确定性和尝试性, 更加适合开放式环境下协同制造装配.

1 $\mathbf{CRQA}_{OVTM}$ Agent 简要描述

1.1 契约 C、处方 R、量化合成 Q 和协作广告 A

"合同" 是制造装配提供者和制造装配请求者之间达成的一种协议. 定义如下:

$$\begin{aligned}&\langle Contr\rangle := (Infor, PROT)\\&\langle Infor\rangle := (CntrtNUM, SerNUM, ReqAg,\\&\qquad ProAg, AuthAg, ExTm, ExpTm)\\&\langle PROT\rangle := (\{CN\}^{+})\end{aligned}$$

$Contr$ 是契约, $Infor$ 是契约信息, 主要有契约号、服务号、请求制造装配 Agent、提供制造装配 Agent、契约管理 Agent、契约执行时间、到期时间等, $PROT$ 是合同履行协议, CN 是规范, 可以是 OB_a^{Con} $(\rho \le \delta|\sigma)$、FB_a^{Con} $(\rho \le \delta|\sigma)$ 和 Pb_a^{Con} $(\rho \le \delta|\sigma)$, 分别是 a 在 σ 条件下, 于截至期限 δ 前有义务、禁止、或权利使 ρ 满足.

"处方" (预估规划) 是对任务按子任务性质进行确定性的任务分解并定义执行流程. 例如, 小型会议安排任务包括宾馆预订、会议门票预订和旅行等不同性质子任务. 其形式化定义如下:

$$\begin{aligned}&\langle Recipe\rangle := (AppTar, ExPlan)\\&\langle AppTar\rangle := \langle EleRef\rangle|\langle TskDs\rangle\\&\langle ExPlan\rangle := \{\langle PlaStep\rangle|(''\mathrm{loop}''\langle PlaStep\rangle)\}^{+}\\&\langle PlaStep\rangle := \{(\leftarrow \{\mathrm{return}|\langle CalOpr\rangle\}\\&\quad \langle Condition\rangle)|(\leftarrow \langle OperaS\rangle[\langle Condition\rangle])|\\&\quad (or\{\leftarrow \langle OperaS\rangle[\langle Condition\rangle]\}^{+})\}^{+}\\&\langle OperaS\rangle := (\{''\mathrm{sequen}''|''\mathrm{concurren}''\}\{\leftarrow\\&\quad \langle CalOpr\rangle[\langle Condition\rangle]\}^{+})\\&\langle CalOpr\rangle := ([\langle \mathrm{Agent}\rangle]\langle tsk-Sub\rangle\\&\quad \{\langle Para-Value\rangle\}^{*})\\&\langle tsk-Sub\rangle := \langle TskDs\rangle\\&\langle TskDs\rangle := (\langle Class\rangle, \langle Constrain\rangle, \langle Para\rangle)\\&\langle Condition\rangle := \langle Frmla-Pred\rangle\end{aligned}$$

其中, $AppTar$ 指处方的任务 (应用对象), $ExPlan$ 是执行方案, $PlaStep$ 是计划步,

$EleRef$ 指向普通、转换、适配和调制服务等, $ParaValue$ 为调用的参数值, $Class$ 是服务范畴, $FrmlaPred$ 是谓词公式.

量化合成是对任务按子任务数量进行确定性的任务分解并定义执行流程. 比如, 宾馆服务需求 (任务) 表达成概念实例为 @$C\_HotelSer$ $SerID$: h_1 $hAmount$: 6 $hPrice$: (@C_Price $Amount$: 105 $Unit$: "RMB"), 其中 $C\_HotelSer$ 为宾馆服务请求概念实例的名称, $SerID$ (服务号)、$hAmount$ (房间数量) 和 $hPrice$ (单价) 是概念实例的属性, 而属性 $hPrice$ 是另一概念实例, 概念实例名称是 C_Price, 其属性有 $Amount$ (单价值) 和 $Unit$ (价格单位). 显然此服务需求需要 6 个房间, 而 agent$_1$ 和 agent$_2$ 各只能提供 3 间, 所以请求的宾馆服务是量上复合服务. 量化合成为在量上不能分配的处方子任务提供可能性分配.

对于 agent 承担的复合服务 S 中的 n 个可分配属性 (输入参数) $\langle Prop_i, Val_i\rangle$, $1 \leq i \leq n$, $Prop_i$ 是属性名称, Val_i 是属性值, m 个代理 agent$_1$, agent$_2$, $\cdots$, agent$_m$ 在属性 $Prop_1$, $Prop_2$, $\cdots$, $Prop_m$ 的分配量分别为向量 $(\boldsymbol{x}_{11}, \boldsymbol{x}_{12}, \cdots, \boldsymbol{x}_{1n})$, $(\boldsymbol{x}_{21}, \boldsymbol{x}_{22}, \cdots, \boldsymbol{x}_{2n})$, $\cdots$, $(\boldsymbol{x}_{m1}, \boldsymbol{x}_{m2}, \cdots, \boldsymbol{x}_{mn})$. 同时 $\mathrm{m}_{ij} \leq \boldsymbol{x}_{ij} \leq \mathrm{M}_{ij}$, 其中 m_{ij}, M_{ij} 是常量, 则满足:

$$\begin{aligned}
\boldsymbol{x}_{11} + \boldsymbol{x}_{21} + \cdots + \boldsymbol{x}_{m1} &= Val_1 \\
\boldsymbol{x}_{12} + \boldsymbol{x}_{22} + \cdots + \boldsymbol{x}_{m2} &= Val_2 \\
&\vdots \\
\boldsymbol{x}_{1n} + \boldsymbol{x}_{2n} + \cdots + \boldsymbol{x}_{mn} &= Val_n
\end{aligned}$$

记 $x_i = (\boldsymbol{x}_{1i}, \boldsymbol{x}_{2i}, \cdots, \boldsymbol{x}_{mi})^{\mathrm{T}}$, $X = [x_1, x_2, \cdots, x_m]^{\mathrm{T}}$, $Val = [Val_1, Val_2, \cdots, Val_n]^{\mathrm{T}}$, $CONST = [1, 1, \cdots, 1]^{\mathrm{T}}$, 可得:

$$X \cdot CONST = Val \tag{1}$$

假设代理 agent$_1$, agent$_2$, $\cdots$, agent$_m$ 所提供的服务在属性 $Prop_1$, $Prop_2$, $\cdots$, $Prop_n$ 之间有线性制约关系 $\boldsymbol{x}_{i1} : \boldsymbol{x}_{i2} : \cdots : \boldsymbol{x}_{in} = \mathrm{c}_{i1} : \mathrm{c}_{i2} : \cdots : \mathrm{c}_{in}$, 其中 c_{i1}, c_{i2}, $\cdots$, c_{in} 是常量. 记 $\boldsymbol{x}_{ij} = \mathrm{c}_{i1} \cdot k_i$, $\mathrm{m}_{ij} \leq k_i \leq \mathrm{M}_{ij}$, 则可得到:

$$\begin{aligned}
\mathrm{c}_{11} \cdot k_1 + \mathrm{c}_{21} \cdot k_2 + \cdots + \mathrm{c}_{m1} \cdot k_m &= Val_1 \\
\mathrm{c}_{12} \cdot k_1 + \mathrm{c}_{22} \cdot k_2 + \cdots + \mathrm{c}_{m2} \cdot k_m &= Val_2 \\
&\vdots \\
\mathrm{c}_{1n} \cdot k_1 + \mathrm{c}_{2n} \cdot k_2 + \cdots + \mathrm{c}_{mn} \cdot k_m &= Val_n
\end{aligned}$$

记 $K = (k_1, k_2, \cdots, k_m)^{\mathrm{T}}$, $c_i = [c_{1i}, c_{2i}, \cdots, c_{ni}]^{\mathrm{T}}$, $C = [c_1, c_2, \cdots, c_m]^{\mathrm{T}}$, $Val = [Val_1, Val_2, \cdots, Val_n]^{\mathrm{T}}$, 则:

$$C \cdot K = Val \tag{2}$$

上述式 (1) 为量上复合服务的量化合成线性规划模型 (任务分配方案). 式 (2) 是式 (1) 的特例. 规划模型中一个解为可行的一个量化服务任务分配方案. 量化合成要用到量化合成模板. 量化合成模板 $\langle Proc - Temp\rangle$ 是量化子服务调度计划的逻辑表示. 可以用 BNF 格式表示如下:

$$\begin{aligned}
&Proc - Temp := \{\langle Pla - Step\rangle | (Loop \\
&\quad \langle Pla - Step\rangle)\}^{+} \\
&\langle Pla - Step\rangle : \{(\leftarrow \mathrm{Return}\langle Condition\rangle) | \\
&\quad (\leftarrow \langle Ser - Set\rangle[\langle Condition\rangle]) | (or \\
&\quad \{(\leftarrow \langle Ser - Set\rangle[\langle Condition\rangle])\}^{+})\}^{+} \\
&\langle Ser - Set\rangle := \langle Service\rangle | (\{''\mathrm{sequen}'' | \\
&\quad ''\mathrm{concurren}''\}\{(\leftarrow \langle Service\rangle[\langle Condition\rangle])\}^{+}) \\
&\langle Condition\rangle := \langle \text{条件表达式} \rangle \\
&\langle Service\rangle := \langle \text{子服务名} \rangle \{\langle \text{属性值} \rangle\}
\end{aligned}$$

其中, 属性值通过任务分配方案得到. 由最优分配方案得到量化子服务 S_1, S_2, $\cdots$, S_m, 用量化子服务实例化量化合成模板, 如图 1 所示, 得到量化合成方案. 图 1 中的 $Condition_1$ 和 $Condition_2$ 决定了得到的最佳方案是并行执行还是顺序执行, 两个条件是由具体的制造装配域决定的.

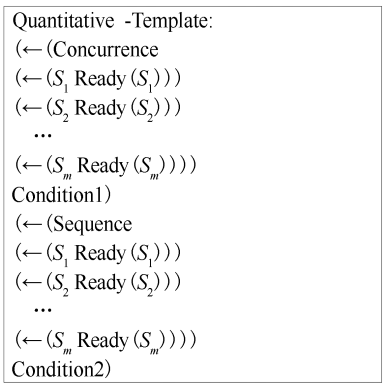

图 1　量化合成

Fig. 1　Quantitative composition

"广告", 就是 $OVTM$ 协作接口广告, 是制造装配商向需求者或中介发送的接口描述, 定义如下:

$$\begin{aligned}
&\langle Advertisement\rangle := (\langle AgentInfo\rangle, \\
&O : \{\langle Ordinary\rangle\}^{*}, V : \{\langle conVersion\rangle\}^{*}, \\
&T : \{adapTor\}^{*}, M : \{\langle Modulator\rangle\}^{*})
\end{aligned}$$

1.2 Agent 协作接口: 普通接口 O、转换接口 V、适配器接口 T 和调制接口 M

CRQA_{OVTM} Agent 模型中的 "普通接口" 是指 Agent 的不经过参数转换就可以直接引用的接口, 它提供普通服务, 学者[12] 称这种服务为能力, 但普通接口可以有更多的可选处理逻辑. 定义如下:

$\langle Ordinary\rangle := \langle Capa\rangle$

$\langle Capa\rangle := (Class, Constrain, Partner\ Constr, Para)$

$\langle Class\rangle := (Category, \{\langle Aspct-N\rangle\ Class-name\}^{+})$

$\langle Aspct-N\rangle := \{''Region''|''ClassGenral''|\ ''ClassPrvsion''\}$

$\langle Constrain\rangle := \langle Constraint\rangle$

$\langle PartnerConstr\rangle := \langle Constraint\rangle$

$\langle Constraint\rangle := \{([''\mathrm{Concurrent}''|''\mathrm{Select}'']\ \{(\langle Pattrn-Cncept\rangle|\langle Constraint>)\}^{+})\}^{+}$

$\langle Pattrn-Cncept\rangle := ([\langle Onto\rangle :]\langle Concept\rangle\ \{\langle Pattrn-Slot\rangle\}^{+}[\langle Frmla-Pred\rangle])$

$\langle Pattrn-Slot\rangle := (nam-Slot, \{value|\ \langle Constraint-Var\rangle|\langle Pattrn-Cncept\rangle\})$

$\langle Constraint-Var\rangle := (name-Var,\ \langle Frmla-Pred\rangle)$

$\langle Para\rangle := (''In-Para :''\ \{\langle Datkind\rangle\}^{+},\ [''Out-Para :''\ \langle Datkind\rangle])$

$\langle Datkind\rangle := \{kind-Bas|[\langle Onto\rangle :]\langle Concept\rangle\}$

其中, $Capa$ 是能力缩写, $Class$ 表示服务的应用域类型, $Constrain$ 表示服务触发时应该满足的条件, $PartnerConstr$ 是协同代理的协作条件, 输入输出参数则用 $Para$ 来表示, $Aspct\text{-}N$ 是侧面, $Onto$ 是应用域本体. $Pattrn\text{-}Cncept$ 是概念模式, $Pattrn\text{-}Slot$ 是槽模式, $nam\text{-}Slot$ 是槽名字, $kind\text{-}Bas$ 是数据基类型, $Datkind$ 是数据类型.

"转换" 是指当请求 Agent 在找不到 "普通接口" 服务提供者时, 服务提供者提供的一种服务. 此接口在 "潜力"[12] 上增加了参数转换路径, 即在触发转换接口内的任务处理逻辑前, 要立即经过服务提供者参数转换才可以进行逻辑处理或者任务处理后需要转换输出参数. "转换" 完成任务精度和可靠性比普通服务低. 其形式化定义如下:

$\langle conVersion\rangle := \langle Capa\rangle$ ($Capa$ 定义同 $Ordinary$ 中 $Capa$ 定义)

并且: $\forall\, requesting$, Matched ($Re.Requesting$, $conVersion$), $\exists\ P \in Para:\ In\text{-}Para$, conversion ($P$) right before invoke $conVersion.\ applying\ logic$; or(and) AFTER invoke $conVersion$, $\exists\ Q \in Para:\ Out\text{-}Para$, $Q=$ conversion ($applyinglogic.\ output$), 其中, $applyinglogic$ 是接口内的任务处理逻辑, $applyinglogic.\ output$ 是任务处理逻辑输出, 不同于接口 $Out\text{-}Para$.

"适配器" 是指当请求 Agent 在找不到 "普通接口" 服务提供者时, 服务提供者提供的另一种服务, 此接口在 "潜力"[12] 上也增加了参数转换路径. 与 "转换" 不同的是, "适配器" 能够进行多种类型的参数转换, 适合参数范围比转换广, 但完成任务精度和可靠性比转换接口完成时的要小. 而且 "适配器" 接口增加了通过参数转换来提供松耦合的合适服务, 提供服务可选手段的范围比潜力[12] 的可选手段范围大 (即是适配器接口完成同一任务的可选路径更多), 可靠性因此比之大大提高. "适配器" 接口定义如下:

$\langle adapTor\rangle := \langle Capa\rangle$ ($Capa$ 定义除 $Para$ 定义外, 同 $Ordinary$ 中 $Capa$ 定义) $\langle Para\rangle := (\{''In-Para :''\ \{\langle Datkind\rangle\}^{+}\}^{+},\ [\{''Out-Para :''\ \{\langle Datkind\rangle\}^{+}\}^{+}])$

并且: $\forall\, requesting$, Matched ($Re.Requesting$, $adapTor$), $\exists\ P \in Para:\ In\text{-}Para$, conversion ($P$) right before invoke $adapTor.applyinglogic$; or(and) AFTER invoke $adapTor$, $\exists Q \in Para:\ Out\text{-}Para$, $Q=$ conversion($applyinglogic.output$), 其中, $applyinglogic$ 是接口内的任务处理逻辑, $applyinglogic.\ output$ 是任务处理逻辑输出, 不同于接口 $Out\text{-}Para$.

"调制" 是指 Agent 可以通过调制转换进行信息处理和提供帮助, 处理的信息包括事实、事件、中间结果等, 调制接口的任务处理逻辑触发前在代理的消息队列中对消息的概念实例 (属性) 进行转换. 通过消息转换进行 "调频", 以便消息适合触发任务处理逻辑, 或者消息部分属性适合触发任务处理逻辑 (但触发的任务处理逻辑仍能提供帮助价值). 调制对服务提供和任务完成情况具有最高的不确定性, 是最具松散的协作接口, 调制接口表明 Agent (可以通过调制转换) 积极提供某种协作信息去帮助请求 Agent 完成任务, 由于可以通过调制转换提供帮助, 比兴趣[12] 提供服务的可选独立路径要多 (包含兴趣提供服务的可选独立路径). 定义如下:

$\langle Modulator\rangle := (Class, PartnerConstr,\ ModulatorInf)$

$\langle ModulatorInf\rangle := \{\langle Pattrn-Cncept\rangle\}^{+}$

[$conversion(ModulatorInf)$before invoke $ModulatorInf$or After invoke $ModulatorInf$]

这里的 [] 表示可选.

1.3 CRQA$_{OVTM}$ Agent 模型描述

CRQA$_{OVTM}$ Agent 模型对传统的 BDI 和推理进行扩展. CRQA$_{OVTM}$ Agent 为 8 元组:

CRQA$_{OVTM}$ Agent := (RE, NKD, RPS, $QSET$, $CONTS$, BDI, PDM, $COMofAg$)

RE: 受 Agent 模型管控的各种可用资源.

NKD: 动态信息和静态知识库.

RPS: 处方集, 是制造装配领域的处方集合.

$QSET$: 量化合成算法集合.

$CONTS$: 合同集, 是代理签订的契约.

BDI: 扩展了的心里模型, 建立了自认识模型 $SELFV$ (对自己的认知)、熟人模型 $PARTM$ (对周围协作 Agent 的认知). $SELFV$ 是自身信息、资源信息和信息接口:

$SELFV := (\langle AgIn\rangle, \langle ResoInf\rangle, \langle CollInf\rangle)$

$PARTM$ 是周围协作代理的广告记录清单:

$PARTM := \{\langle Advertisement\rangle\}^{+}$.

PDM: 负责规划、调度任务, 依据 BDI 心理模型对协作的处方和量化合成算法进行推理, 并遵循合同约束.

$COMofAg$: 负责 Agent 之间的通讯.

2 CRQA$_{OVTM}$ Agent 下的混合制造装配协作

2.1 CRQA$_{OVTM}$ Agent 的协作模式

在 CRQA$_{OVTM}$ Agent 系统里, 建立的混合协作关系包括普通二元协作 OBC (Ordinary binary collaboration)、转换二元协作 VBC (Conversion binary collaboration)、适配性二元协作 TBC (adaptor binary collaboration) 和调制性协作关系 MBC (Modulator binary collaboration) 4 种, 如图 2 所示.

OBC 可以如下定义:

$OBC := \{\ (Re,\ Pr) \,|\, \mathrm{Kind}\,(Re),\ \mathrm{Kind}\,(Pr) = \mathrm{Agent} \bigwedge \exists\, Ordi\,(Ordi \in Pr.Ordinary \bigwedge \mathrm{Matched}\,(Re.Reques\ ting,\ Ordi) \bigwedge \mathrm{InvoReq}\,(Re,\ Pr,\ Re.Repuesting))\}$, 其中, Matched 为匹配, InvoReq 为执行请求. OBC 的特点是请求的服务不需经过参数转换就可以直接引用执行.

VBC 为如下定义:

$VBC := \{\ (Re,\ Pr) \,|\, \mathrm{Kind}\,(Re),\ \mathrm{Kind}\,(Pr) = \mathrm{Agent} \bigwedge \exists\ conV\,(conV \in Pr.conVersion \bigwedge \mathrm{Matched}\,(Re.Requesting,\ conV) \bigwedge \mathrm{Convertand}\ \mathrm{InvoReq}(Re,\ Pr,\ Re.Repuesting))\}$

VBC 的特点是完成任务可靠性比普通协作接口低, 但松耦合程度比普通接口高.

寻求 Agent 可以通过 TBC 关系与具有匹配该请求任务的 $adapTor$ 的提供 Agent 建立 "适配性二元协作" (简称 TBC). TBC 为如下定义:

$TBC := \{\ (Re,\ Pr) \,|\, \mathrm{Kind}\,(Re),\ \mathrm{Kind}\,(Pr) = \mathrm{Agent} \bigwedge \exists\ adapTor\,(adapTor \in Pr.adapTor \bigwedge \mathrm{Matched}\,(Re.Requesting, adapTor) \bigwedge \mathrm{AttemReq}\,(Re,\ Pr,\ Re.Repuesting))\}$

TBC 具有不确定性, 提供 Agent 要经过适配性转换提供服务.

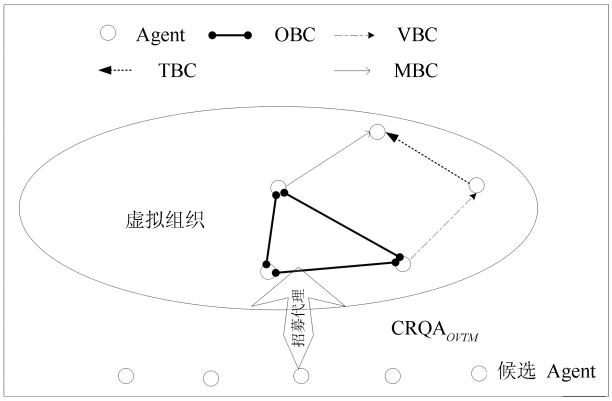

图 2　CRQA$_{OVTM}$ Agent 下混合协作关系示意图

Fig. 2　Diagrammatic sketch of hybrid cooperations under CRQA$_{OVTM}$ Agent

当提供方 Agent 通过 "调制" 提供服务时, 寻求 Agent 便可以发送消息给该 Agent, 建立 "调制性协作关系" (缩写 MBC). MBC 为如下定义:

$MBC := \{\ (Re,\ Pr) \,|\, \mathrm{Kind}\,(Re),\ \mathrm{Kind}(Pr) = \mathrm{Agent} \bigwedge \exists\ Modulator,\ \exists\ Infom,\ \mathrm{beMemb}\,(Modulator, Pr.Modulator) \bigwedge Infom \in Re.KB \bigwedge \mathrm{Matched}\,(Infom,\ Modulator) \bigwedge \mathrm{convert}\ \mathrm{Messg}\,(Re,\ Pr,\ Infom))\}$, 且 KB 是拥有的知识, beMemb 是成员关系谓词, $convertMessg$ 是需要消息参数转换的消息通讯.

2.2 CRQA$_{OVTM}$ Agent 的协同机理与协作合理性分析证明

1) CRQA$_{OVTM}$ Agent 协同机理概括

协作机理如图 3 所示, 其大致步骤概括如下:

步骤 1. 当请求任务到达时, 根据自认识模型 SELFV 判断自身能否独立解决任务. 假如能, 则独立完成任务并将结果还回, 不需要其他 Agent 参与.

步骤 2. 根据熟人模型与中介机构检查是否存在某个 Agent 的普通接口可以匹配要执行的任务. 设 $Set\_IOrdinary = \{Pr.Ordi \,|\, \mathrm{kind}\ \ (Pr) = \mathrm{Agent} \bigwedge \mathrm{BELI}_{SELFV}\,(Pr.Ordi \in Pr.Ordinary \bigwedge\ (\mathrm{Matched}\,(SELFV.Requesting, Pr.Ordi)\ \ \mathrm{or}$

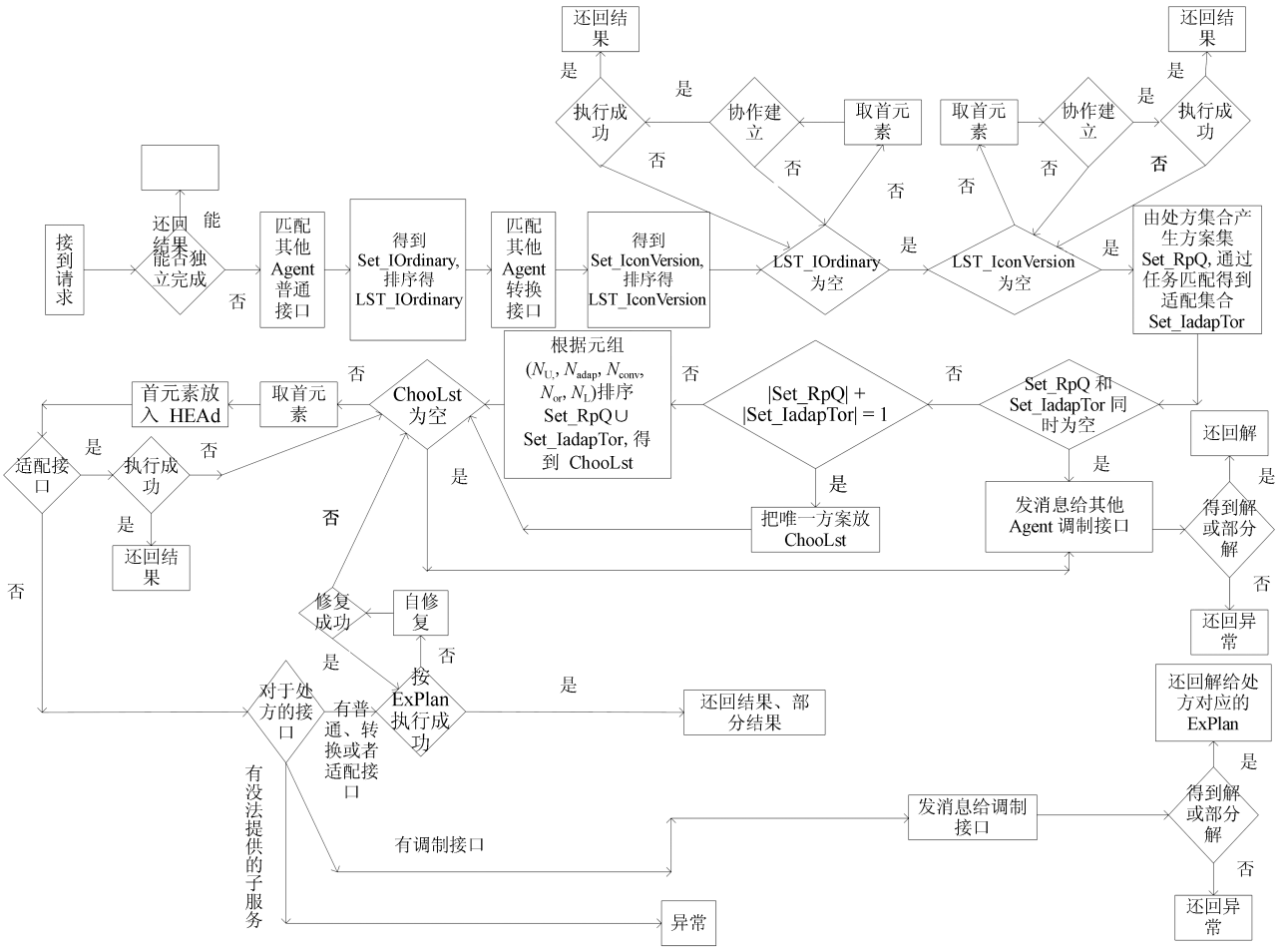

图 3 协作机理

Fig. 3 Cooperation mechanism

Matched (*quantization* (*SELFV.Requesting*), *Pr.Ordi*))}. *Quantization* 为量化分解. 排序 *Set_IOrdinary* 中的元素 (可以根据协同情景以及自身意愿进行排序, 每一个量化方案中的子任务通过匹配得到接口集, 接口集作为一个元素进行排序) 产生 *Lst_IOrdinary*.

步骤 3. 根据熟人模型与中介机构检查是否存在某个 Agent 的转换接口可以匹配要执行的任务. 设 $Set\_IconVersion = \{Pr.conV \mid \text{kind}(Pr) = \text{Agent} \bigwedge \text{BELI}_{SELFV}(Pr.conV \in Pr.conVersion \bigwedge (\text{Matched}(SELFV.Requesting, Pr.conV) \text{ or } \text{Matched}(quantization(SELFV.Requesting), Pr.conV))\}$. 排序 *Set_IconVersion* 中的元素 (可以根据协同情景以及自身意愿进行排序. 每一个量化方案中的子任务通过匹配得到接口集, 接口集作为一个元素进行排序) 产生 *Lst_IconVersion*.

步骤 4. 假如 *Lst_IOrdinary* 为空, 则转向步骤 5. 取 *Lst_IOrdinary* 的首元素 *Pr.Ordi*, 与提供代理建立普通二元协作. 假如执行成功, 则转向步骤 11; 失败就再次执行步骤 4.

步骤 5. 假如 *Lst_IconVersion* 为空, 则转向步骤 6. 取 *Lst_IconVersion* 的首元素 *Pr.conVersion*, 与提供代理建立确定转换二元关系. 假如执行成功, 则转向步骤 11; 失败就再次转向步骤 5.

步骤 6. 让处方的集合 $Set\_Rep = \{rcp \mid \text{BELI}_{SELFV}(rcp \in SELFV.RPS \bigwedge \text{Matched}(SELFV.Requesting, rcp.AppTar))\}$, *Set_Rep* 中每个处方与量化合成算法产生对应处方的方案集, 每个处方对应的方案集的并得到总方案集 *Set_RpQ*, 并设与任务匹配的适配集合 $Set\_IadapTor = \{Pr.adap \mid \text{kind}(Pr) = \text{Agent} \bigwedge \text{BELI}_{SELFV}(Pr.adap\ Pr.adapTor\ \text{Matched}(SELFV.Requesting, Pr.adap))\}$. 假如 *Set_RpQ* 为空同时 *Set_IadapTor* 为空, 转向步骤 10; 假如 $|Set\_RpQ|+|Set\_IadapTor| = 1$, 则此执行方案设为 *ChooLst* 中唯一方案, 并转向步骤 8.

步骤 7. 逐个为 Set_RpQ 集合的每个方案计算其中可由代理自己提供、通过其他代理的普通接口提供、转换接口提供、适配接口提供和无法提供的子服务数目, 产生 $(N_U,\ N_{adap},\ N_{conv},\ N_{or},\ N_L)$. 其中, N_U 表示无法提供的子服务数目, N_{adap} 表示通过适配接口提供的子服务数目, N_{conv} 表示通过转换提供的子服务数目, N_{or} 表示通过普通接口提供的子服务数目, N_L 表示通过代理自己提供的子服务数目. 对 $Set\_IadapTor$ 中的元素, 其 $(N_U,\ N_{adap},\ N_{conv},\ N_{or},\ N_L) = (0, 1, 0, 0, 0)$. 排序集合 $Set\_RpQ \cup Set\_IadapTor$ 中的方案, 产生 $ChooLst$, 并按 $(N_U,\ N_{adap},\ N_{conv},\ N_{or},\ N_L)$ 字典序对其排序.

步骤 8. 假如 $|ChooLst|$ 为空, 则转向步骤 10. 否则取 $ChooLst$ 的第一个元素放入 $Head$. 假如 $Head$ 是处方方案, 则遵照方案的 $ExPlan$ 执行. 如果 $Head$ 是适配器, 则建立适配性协作. 同时, 对匹配上 $PARTM$ 列表中的调制接口的概念实例, 给有调制的代理选择性地发消息.

步骤 9. 若方案执行失败, 则通过协作关系的自修复、自演化来应对失败, 实现系统稳定. 从两个方面进行: 从契约监督执行的角度, 促进 Agent 检查产品是否符合质量要求 (可人工检查), 如不达要求, 则按照制裁规范实施制裁: 从协同演化的角度, 虚拟组织 (Virtual organization, VO) 发起者根据契约执行状态, 如果状态显示质量未达标, 则违约事件就促发相应的演化活动, 包括修改服务结果 (更换零部件)、更改服务提供方、修改服务提供时间、或者更改处方方案分支 (重新装配)、甚至解散 VO 而重新组建新的 VO. 以后继续监视契约执行, 直到最终达标或者未达标. 执行结果成功则转向步骤 11; 最终不成功则重新转向步骤 8.

步骤 10. 对匹配上 $PARTM$ 列表中的调制接口的概念实例, 给有调制接口的代理选择性地发消息.

步骤 11. 当得到完全解或部分解, 返回解; 若执行最终不成功, 返回异常出错信息.

2) 合理性分析证明

定理 1. 假设完成某任务 $TASK$ 有相互独立的不同的可选方案 $\{A_1,\ A_2,\ A_3,\ \cdots,\ A_n\}$, $\{B_1,\ B_2,\ B_3,\ \cdots,\ B_n\}$, 如果方案 A_j 的成功概率 $P(A_j)$ 大于或等于方案 B_j 的成功率 $P(B_j)$ $j = 1, \cdots, n$, 则方案集 $\{A_1,\ A_2,\ A_3,\ \cdots,\ A_n\}$ 成功率 $P(A_1,\ A_2,\ A_3,\ \cdots,\ A_n)$ 大于或等于方案集 $\{B_1,\ B_2,\ B_3,\ \cdots,\ B_n\}$ 的成功概率 $P\,(B_1,\ B_2,\ B_3,\ \cdots,\ B_n)$.

证明. 依据方案集 $\{A_1, A_2, A_3, \cdots, A_n\}$ 执行任务 $TASK$ 失败时的概率是 $\prod_{i=1}^{n}\big(1 - P(A_i)\big)$, 依据方案集 $\{B_1,\ B_2,\ B_3,\ \cdots,\ B_n\}$ 执行任务 $TASK$ 失败时的概率是 $\prod_{i=1}^{n}\big(1 - P(B_i)\big)$, 由于 $P(A_i) \geq P(B_i)$, 则 $\prod_{i=1}^{n}\big(1 - P(A_i)\big) \leq \prod_{i=1}^{n}\big(1 - P(B_i)\big)$, 故 $1 - \prod_{i=1}^{n}\big(1 - P(A_i)\big) \geq 1 - \prod_{i=1}^{n}\big(1 - P(B_i)\big)$, 所以结论成立. □

定理 2. 假设完成某任务 $TASK$ 有相互独立的可选方案 $A_1,\ A_2,\ A_3,\ \cdots,\ A_n$ 和相互独立的可选方案 $A_1,\ A_2,\ A_3,\ \cdots,\ A_n,\ B_1,\ B_2,\ B_3,\ \cdots,\ B_n$, 则 $P(A_1,\ A_2,\ A_3,\ \cdots,\ A_n,\ B_1,\ B_2,\ B_3,\ \cdots,\ B_n) \geq P(A_1,\ A_2,\ A_3,\ \cdots,\ A_n)$.

证明. $P(A_1,\ A_2,\ A_3,\ \cdots,\ A_n,\ B_1,\ B_2,\ B_3,\ \cdots,\ B_n) \geq P(A_1,\ A_2,\ A_3,\ \cdots,\ A_n) + (1 - P(A_1,\ A_2, A_3, \cdots, A_n)) \cdot P(B_1, B_2, B_3, \cdots, B_n) \geq P(A_1,\ A_2,\ A_3,\ \cdots,\ A_n)$. □

由以上定理, 我们不难得出以下结论:

结论 1. CRQA$_{OVTM}$ Agent 有比紧耦合类型 Agent 更高任务完成率.

证明. 紧耦合类型 Agent 只有包含步骤 1 (本地方案) 和步骤 2 (普通接口方案) 的相互独立的任务完成方案, 而 CRQA$_{OVTM}$ Agent 有包含步骤 1、步骤 2、步骤 3 (转换接口方案)、步骤 6(处方、量化合成和适配性方案) 和步骤 10 (调制方案) 的相互独立方案. 根据定理 2, 结论 1 成立. □

结论 2. CRQA$_{OVTM}$ Agent 有比 RPA_{CTI} Agent 更高任务完成率.

1) CRQA$_{OVTM}$ Agent 的接口成功率高于 RPA_{CTI} Agent 的接口成功率, 即是 $P(O) \geq P(C)$, $P(V) \geq P(potentiality)$, $P(adapTor) \geq P(potentiality)$, $P(M) \geq P(I)$.

证明. 根据定义, 由于普通接口有比能力接口更多的完成任务的可选路径, 转换接口有比潜力接口更多的完成任务的可选路径, 适配接口有比潜力接口更多的完成任务的可选路径, 调制接口有比兴趣接口更多的完成任务的可选路径, 根据定理 2, 结论成立. □

2) CRQA$_{OVTM}$ Agent 有比 RPA_{CTI} Agent 更高任务完成率.

证明. RPA$_{CTI}$ Agent 有包含步骤 1 (本地方案)、步骤 2 (能力方案, 即普通接口方案)、步骤 6 (处方、潜力方案) 和步骤 10 (兴趣方案) 的相互独立方案, 而 CRQA$_{OVTM}$ Agent 有包含步骤 1 (本地方案)、步骤 2 (普通接口方案)、步骤 3 (转换接口方案)、步骤 6 (处方、量化合成和适配性方案) 和步骤 10 (调制方案) 的相互独立方案, 比 RPA$_{CTI}$ Agent 多了转换接口方案和量化合成方案集, 同时适配性方案有比潜力方案更高的成功率 (根据 1)), 调制方案有比兴趣方案更高的成功率 (根据 2)), 结合定理 1 和定理 2, 结论 2 成立. □

3 举例与实验分析

3.1 举例

为应对矿难, 要制造装配 20 000 个智能矿工安全帽, 要满足: 1) 大小为中型; 2) 配有低频无线发射器进行视频语音通讯; 3) 有卡口 LED 灯泡供照明. 表示 1 个智能矿工安全帽的任务的条件约束部分可以用以下概念模式: (@$SecuCap$ $Size$: "Middle", $Attch1$: ?x(@$Flshlit$ $kind$: "Led" $Scle$: "Large" $intface$ "bayonet"), $Attch2$: ?y (@$Radio$ $Frequency$: "Low")

制造装配任务交给了客户代理, 如图 4 (每类接口定义了同任务的多个可选路径). 由于业务繁多, 客户代理软件刚签订矿工安全帽合同又接到其他商品订单, 这些订单将要占用 20 000 个智能矿工安全帽所需部分制造资源. 此时客户代理不能独立完成安全帽制造. 依据熟人模型 $PARTM$, 客户代理软件发现 Agwhole_1 和 Agwhole_2 的转换接口能匹配概念模式 (@$SecuCap$), 但都只能制造装配 10 000 个智能矿工安全帽, 此时出现结构失配 (随着制造业务的增多, 占有相关资源越多, 制造时间越紧, 此特性越易出现). 于是客户代理进行量化合成 (依据第 2.2 节步骤 3), 把 10 000 个智能矿工安全帽制造装配任务给 Agwhole_1, 另 10 000 个智能矿工安全帽制造装配任务给 Agwhole_2.

出于盈利目的, Agwhole_1 同样刚签订矿工安全帽合同又接了其他商品订单, 检查自身资源发现资源不足, 没有能力解决安全帽制造任务. 由于可

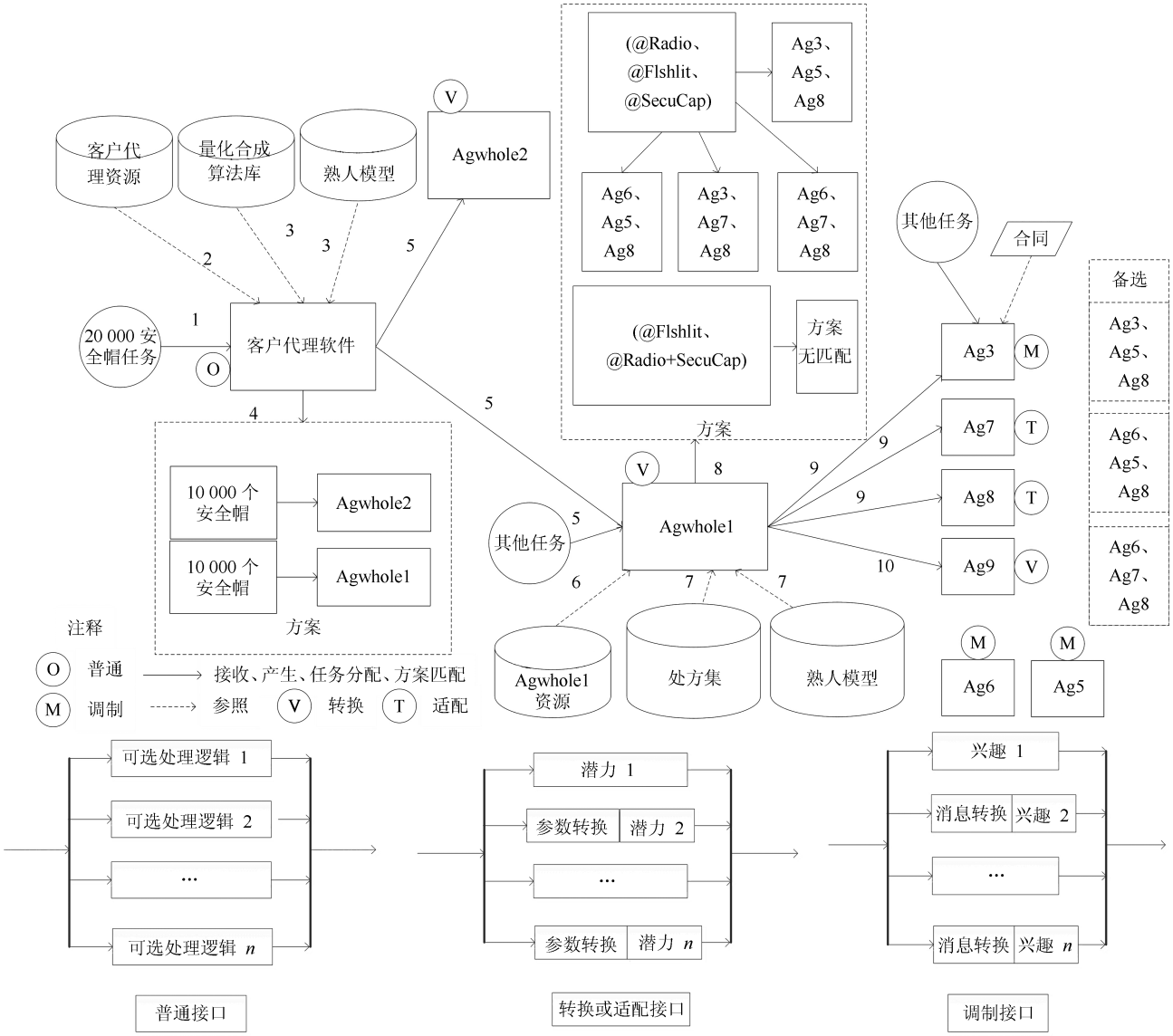

图 4 应用实例

Fig. 4 Applying case

用资源已经饱和, $\mathrm{Agwhole}_2$ 经过决策分析不再接收其他制造任务, 以保证自己独立完成客户代理软件交给的安全帽制造任务. 由于合同时间约束, 此时 $\mathrm{Agwhole}_1$ 进行任务分解. 通过第一处方分解请求的任务, 得到子任务 @*Radio*、@*Flshlit*、@*SecuCap*, 通过第二处方得到子任务 @*Flshlit*、@*Radio*+*SecuCap* (无线与帽连). 由于短期没有 Agent 的普通接口可以匹配子任务 @*Radio*、@*Flshlit*、@*SecuCap* (普通接口由于是确定性服务, 业务繁多, 所需资源已经占用, 无法提供服务; 或者由于业务繁多, 其性能已经严重下降), 此时协同制造过程中出现 "结构失配" 和 "工艺革新" 的特性. 这时只有通过匹配本文中定义的接口才能解决 "结构失配" 和 "工艺革新" 的问题. 依据熟人模型 $PARTM$, 通过匹配任务约束信息, 发现 Ag_3 和 Ag_6 分别对 (@*Radio*) 能提供调制服务, Ag_5、Ag_7 分别对 (@*Flshlit*) 能提供调制服务和适配服务, Ag_8 对 @*SecuCap* 能提供适配服务. 于是根据第一处方分配方案, 可以得到 4 种子任务分配方案, 即是子任务 (@Radio、@Flshlit、@*SecuCap*) 分别分配给 (Ag_3、Ag_7、Ag_8)、(Ag_6、Ag_7、Ag_8)、(Ag_6、Ag_5、Ag_8)、(Ag_3、Ag_5、Ag_8). 由于短期没有 Agent 的普通接口可以匹配子任务 @*Radio*、@*Flshlit* + *SecuCap*, 再次出现 "结构失配" 和 "工艺革新" 的特性. 由于暂时没有找到 Agent 的接口可以匹配子任务 @*Radio* + *SecuCap*, 同时由于资源所限, $\mathrm{Agwhole}_1$ 不能自己完成子任务 @*Radio* + *SecuCap*, 因此无法提供服务完成子任务 @*Radio* + *SecuCap*. 依据第 2.2 节中机理对方案的排序, $\mathrm{Agwhole}_1$ 选择将子任务 (@*Radio*、@*Flshlit*、@*SecuCap*) 分配给 (Ag_3、Ag_7、Ag_8), 分别和 Ag_3、Ag_7、Ag_8 建立调制性协作、适配性协作和适配性二方协作.

Ag_8 收到 @*SecuCap* 的消息后, 发现自己有足够资源, 于是完成安全帽制作. Ag_3 收到 (@*Radio*) 的消息后, 它意识自己只能制造高频的无线收发器, 并且通过频率转换把高频转换成所要的低频出现故障. 尝试进行修复失败. 于是 $\mathrm{Agwhole}_1$ 选择将子任务 (@*Radio*、@*Flshlit*、@*SecuCap*) 分配给 (Ag_6、Ag_7、Ag_8), 即是把 (@*Radio*) 子任务分配给 Ag_6, 并和 Ag_6 建立调制性二方协作.

Ag_6 通过调制知识最终完成了低频无线收发器的制作, 制造期间也跟其他任务产生冲突, 但由于合同约束, 最终将制造结果返回给 $\mathrm{Agwhole}_1$. Ag_7 通过适配知识发现自己只能制造螺口的灯泡, 并且通过接口转换能把螺口转换成卡口灯泡, 最后将结果返回给 $\mathrm{Agwhole}_1$.

当 $\mathrm{Agwhole}_1$ 收到制造结果后, 发现没有 Agent 的普通接口可以匹配 Ag_6、Ag_7 和 Ag_8 的产品装配需求, 再次出现 "结构失配" 和 "工艺革新" 的问题. 但经搜索, 发现 Ag_2、Ag_4 和 Ag_8 的产品符合产品装配 Ag_9 的转换接口, 将三个独立的产品发给 Ag_9. Ag_9 一般不接收产品, 而是自己生产产品进行装配, 但是进行服务参数转换后可以接收产品进行装配, 将接收的安全帽、无线收发器和灯泡装配后, 把成品还回给 $\mathrm{Agwhole}_1$.

客户代理软件收到 $\mathrm{Agwhole}_1$ 和 $\mathrm{Agwhole}_2$ 还回的结果后, 最终完成了制造装配两个智能矿工安全帽的任务, 并把结果还回给客户.

3.2　实验和结果分析

CRQA_{OVTM} Agent 模型和系统引入参数松耦合, 提出确定性转换和非确定性参数转换, 最大限度地提高协作的松耦合度和可靠性. 以前 Agent 系统只是通过紧耦合或者集中控制方式进行协同制造装配, 或者只对应用域进行松耦合, 忽略了参数松耦合. 我们对以前 RPA_{CTI} Agent[12] 协同系统与 CRQA_{OVTM} Agent 协同系统解决任务的成功率进行了比较. 我们分别在样本任务数都是 500, 而任务约束项数目 $|Con\_Ites|$ 分别为 $4\sim7$, $10\sim12$, $22\sim25$ 和 $30\sim32$, 两个系统拥有同样的资源, 都只有 20 个 Agent 协作的情况下进行了仿真实验.

1) 人工设定接口概率

由于服务的运行时推理、集成的对外提供以及范围大的实现不同任务的服务由确定人群 (确定技能和知识点) 编写, 导致服务的适用性精度和完成的可靠性下降. CRQA_{OVTM} Agent 通过在 RPA_{CTI} Agent 的不确定性接口中增加不确定性参数转换模块去完成相同任务, 就提高了接口的可靠性 (也有力地应对了 "结构失配" 与 "工艺革新" 问题). 例如, CRQA_{OVTM} Agent 的多参数适配器在 RPA_{CTI} Agent 的直接完成任务的潜力模块的基础上增加了参数转换间接完成任务模块, 见图 4. 假如要完成螺口灯泡制作, 路径 1: CRQA_{OVTM} Agent 的多参数适配器可以在具备的 RPA_{CTI} Agent 潜力模块上直接完成螺口灯泡制作; 路径 2: 也可以通过适配器具备的潜力模块完成卡口灯泡制作, 再通过参数转换把卡口转换成螺口. 如果路径 1 的可靠性是 70 %, 路径 2 的可靠性是 50 %, 则多参数适配器的可靠性是 $1-(1-70\,\%)(1-50\,\%)=85\,\%$, 明显比 RPA_{CTI} Agent 的潜力 70 % 可靠性高.

仿真屏蔽了制造实现细节, 仅仅仿真制作的可靠性. 在这里我们通过 C++ 下的 *rand* 函数来实现

仿真某个接口以某个概率完成某任务制造 (函数以某概率还回成功). $rand(\cdot)$ 可以返回一个从 0 到最大随机数的任意整数. 如果要产生 $a \backsim a+n$ 的 n 个整数中的随机整数, 可以表示为: $a + rand(\cdot)\,\%\,n$.

基于随机思想, 设置接口有 80 % 的制造成功概率, 采取 $rand(\cdot)$ 函数实现如下:

if $(21 == 20 + rand(\cdot)\,\%10 \,\text{or}\, 22 == 20 + rand(\cdot)\,\%10)$

{//$20 + rand()\,\%10$ 可以产生在随机区间 $20 \sim 30$ 的 10 个不同的随机整数. $20 == 20 + rand(\cdot)\,\%10$ or $22 == 20 + rand(\cdot)\,\%10$ 只满足其中的二个, 所以这里的概率 20 %, 这里的代码只写一句话 return false}

else

{//80 % 的概率成功完成制造任务, 这里的代码只写一句话 return true}

本实验设定的接口成功概率是固定值, 也可以设为可变 (怎样编程只是细节, 不再叙述).

2) 任务成功率的计算

按照以上思想设定 RPA_{CTI} Agent 每个接口完成对应样本任务的概率, 设定 CRQA_{OVTM} Agent 每个接口完成对应样本任务的概率 (高于对应 RPA_{CTI} Agent 接口的概率), 或者设定 RPA_{CTI} Agent 每个接口完成样本任务对应子任务的概率, 设定 CRQA_{OVTM} Agent 每个接口完成样本任务对应子任务的概率 (高于对应 RPA_{CTI} Agent 接口的概率). 约定某一个样本任务对应的所有子任务接口都 return true 时, 这个样本任务才执行成功. 对于某个约束项目数, 每类 Agent 模型中的所有成功样本任务数 / $500 \times 100\,\%$ 就是本 Agent 模型完成任务的一次成功率. 为了接近饱和实验, 取 10 000 次成功率的平均作为最后的成功率.

3) 方案的人工设定

对每个任务 $TASK_i$, i 等于 1 到 500, 设定其对应的方案为

$\{[Ag_{m1} \cdot S_{m1}(POSS_{m1})],\ [Ag_{m2} \cdot S_{m2}(POSS_{m2})], \cdots, [Ag_{mn} \cdot S_{mn}(POSS_{mn})]\}^*$, n 为约束项 (子任务数), 可以等于 $4 \sim 7$, $10 \sim 12$, $22 \sim 25$, $30 \sim 32$; Ag_{mx} 为 20 个 RPA_{CTI} Agent 模型 (或者 CRQA_{OVTM} Agent 模型, 同一方案只能一种模型) 中一个 Agent, 对于 RPA_{CTI} Agent 模型, S_{mx} 为 Ag_{mx} 中的能力或者潜力或者兴趣接口 (对于 CRQA_{OVTM} Agent 模型, 为普通接口或者转换接口或者适配接口或者调制接口), 这里的接口仅仅以实验设定的概率还回成功或者失败, $POSS_{mx}$ 为完成子任务的概率, x 取 $1 \sim 20$, [·] 为可选, 当 {·} 中没有元素时表明方案为空, 意为任务找不到方案, 当 {·} 中连续出现两个 "," 时, 表明任务不可解; $*$ 表明任务 $TASK_i$ 对应 0 到多个人工排序方案.

例如: 对应约束项数为 7 时:

任务 1 有 7 个子任务, 对应的 RPA_{CTI} Agent 模型的一个方案设定及执行情况为 {Agent_1 的第一个潜力接口 (执行子任务 1, 概率函数还回 true), Agent_2 的第三个潜力接口 (执行子任务 2, 概率函数还回 true), Agent_3 的第一个兴趣接口 (执行子任务 3, 概率函数还回 true), Agent_6 的第一个潜力接口 (执行子任务 4, 概率函数还回 true), Agent_8 的第一个潜力接口 (执行子任务 5, 概率函数还回 true), Agent_9 的第三个兴趣接口 (执行子任务 6, 概率函数还回 true), Agent_{17} 的第一个能力接口 (执行子任务 7, 能力函数还回 true, 设定能力 100 % 成功)}, 由于每个子任务都执行成功, 任务 1 执行成功. 同理, 对应的 CRQA_{OVTM} Agent 模型的一个方案也可以具体设定并有对应执行成功时情况.

4) 实验前的满足条件

4.1) RPA_{CTI} Agent 模型的任何一个任务的方案中的任何一个接口, 必然在 CRQA_{OVTM} Agent 模型中找到对应任务的方案中的对应接口并且满足 CRQA_{OVTM} Agent 的接口成功概率高于对应 RPA_{CTI} Agent 接口成功概率. 例如: RPA_{CTI} Agent 模型的 $TASK_i$ 中某个方案有元素潜力, 则 CRQA_{OVTM} Agent 模型的 $TASK_i$ 某个方案必然有适配, 适配的成功率高于潜力.

4.2) 随着对应约束项 (子任务) 增加, 两个模型对应的设定方案中出现的不能由 Agent 匹配到的子任务数都将越来越多, 但是 CRQA_{OVTM} Agent 有接口具备参数转换功能, 更容易匹配到任务 (松耦合性更高), 因此虽然随着对应约束项 (子任务) 的增加, 设定的方案中出现不能由 Agent 匹配到的子任务将越来越多, 但是比 RPA_{CTI} Agent 设定的方案, 这种不能匹配的子任务数上升趋势缓慢.

4.3) 由于 Agent 心智模型 (如: 方案排序) 已经人工设定, 因此方案中的 Agent 只是编号.

5) 实验数据及结论

本文所涉及的两类 Agent 接口部分数据设定如表 1 所示. 表中, {·} 里面的数据分别表示任务号、子任务号、对应的接口成功完成子任务的概率. 一个类型的接口可以完成多项任务.

表 2 是人工设定的任务解决方案. {} 里面的数据是完成子任务的接口集合. 例如任务 1 的 CRQA_{OVTM} Agent 方案是 {$\text{AGENT}_3.M$ (62 %), $\text{AGENT}_2.T$ (84 %), $\text{AGENT}_1.O$, AGENT_4.O}, 意思是任务 1 的子任务 1 由 AGENT_3 的调制接口完成, 成功概率是 62 %, 子任务 2 由 AGENT_2 的适配接口完成成功概率是 84 %,

表 1　任务约束项数为 4～7 时两类模型的接口成功概率数据

Table 1　Two model interface success rate data under task constraints 4～7

接口代理	能力	潜力	兴趣	普通接口	转换接口	适配接口	调制接口	接口代理
Agent_1	{1, 3, 1}	{2, 3, 75 %}	{61, 3, 50 %}	{1, 3, 1}	{3, 1, 93 %}	{2, 3, 86 %}	{61, 3, 60 %}	AGENT_1
	{5, 1, 1}	{4, 1, 75 %}	{83, 1, 50 %}	{5, 1, 1}	{6, 3, 93 %}	{4, 1, 86 %}	{83, 1, 60 %}	
	{11, 1, 1}	{12, 1, 75 %}	{95, 1, 50 %}	{11, 1, 1}	{7, 2, 93 %}	{12, 1, 86 %}	{95, 1, 60 %}	
	{15, 1, 1}	{16, 1, 75 %}	{97, 1, 50 %}	{15, 1, 1}	{8, 2, 93 %}	{16, 1, 86 %}	{97, 1, 60 %}	
	{21, 2, 1}	{22, 2, 75 %}	{75, 2, 50 %}	{21, 2, 1}	{9, 3, 93 %}	{22, 2, 86 %}	{75, 2, 60 %}	
	{51, 3, 1}	{52, 3, 75 %}	{87, 3, 50 %}	{51, 3, 1}	{10, 2, 93 %}	{52, 3, 86 %}	{87, 3, 60 %}	
Agent_2	{2, 2, 1}	{1, 2, 73 %}	{61, 2, 55 %}	{2, 2, 1}	{61, 3, 96 %}	{1, 2, 84 %}	{61, 2, 64 %}	AGENT_2
	{4, 2, 731}	{5, 2, 73 %}	{83, 2, 55 %}	{4, 2, 1}	{83, 1, 96 %}	{5, 2, 84 %}	{83, 2, 64 %}	
	{12, 3, 1}	{11, 3, 73 %}	{95, 3, 55 %}	{12, 3, 1}	{95, 1, 96 %}	{11, 3, 84 %}	{95, 3, 64 %}	
	{16, 4, 1}	{15, 4, 73 %}	{97, 4, 55 %}	{16, 4, 1}	{97, 1, 96 %}	{15, 4, 84 %}	{97, 4, 64 %}	
	{22, 1, 1}	{21, 1, 73 %}	{75, 1, 55 %}	{22, 1, 1}	{75, 2, 96 %}	{21, 1, 84 %}	{75, 1, 64 %}	
	{52, 1, 1}	{51, 1, 73 %}	{87, 1, 55 %}	{52, 1, 1}	{87, 3, 96 %}	{51, 1, 84 %}	{87, 1, 64 %}	
Agent_3	{61, 1, 1}	{2, 1, 76 %}	{1, 1, 52 %}	{61, 1, 1}	{13, 2, 95 %}	{2, 1, 84 %}	{1, 1, 62 %}	AGENT_3
	{83, 3, 1}	{4, 3, 76 %}	{5, 3, 52 %}	{83, 3, 1}	{14, 2, 95 %}	{4, 3, 84 %}	{5, 3, 62 %}	
	{95, 2, 1}	{12, 2, 76 %}	{11, 2, 52 %}	{95, 2, 1}	{17, 3, 95 %}	{12, 2, 84 %}	{11, 2, 62 %}	
	{97, 2, 1}	{16, 2, 76 %}	{15, 2, 52 %}	{97, 2, 1}	{18, 4, 95 %}	{16, 2, 84 %}	{15, 2, 62 %}	
	{75, 3, 1}	{22, 3, 76 %}	{21, 3, 52 %}	{75, 3, 1}	{19, 1, 95 %}	{22, 3, 84 %}	{21, 3, 62 %}	
	{87, 2, 1}	{52, 2, 76 %}	{51, 2, 52 %}	{87, 2, 1}	{20, 1, 95 %}	{52, 2, 84 %}	{51, 2, 62 %}	
Agent_4	{1, 4, 1}	{2, 4, 78 %}	{61, 4, 51 %}	{1, 4, 1}	{23, 4, 94 %}	{2, 4, 85 %}	{61, 4, 63 %}	AGENT_4
	{5, 4, 1}	{4, 4, 78 %}	{83, 4, 51 %}	{5, 4, 1}	{24, 4, 94 %}	{4, 4, 85 %}	{83, 4, 63 %}	
	{11,5,1}	{12,5,78 %}	{95,5,51 %}	{11,5,1}	{25,5,94 %}	{12,5,85 %}	{95,5,63 %}	
	{15, 3, 1}	{16, 3, 78 %}	{97, 3, 51 %}	{15, 3, 1}	{26, 3, 94 %}	{16, 3, 85 %}	{97, 3, 63 %}	
	{21, 4, 1}	{22, 4, 78 %}	{75, 4, 51 %}	{21, 4, 1}	{27, 4, 94 %}	{22, 4, 85 %}	{75, 4, 63 %}	
	{51, 4, 1}	{52, 4, 78 %}	{87, 4, 51 %}	{51, 4, 1}	{28, 4, 94 %}	{52, 4, 85 %}	{87, 4, 63 %}	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

表 2　任务约束项数为 4～7 时方案

Table 2　Schemes under task constraints 4～7

任务	约束项数	RPA_{CTI} Agent 方案	CRQA_{OVTM} Agent 方案	说明
任务 1	$4 \in 4 \sim 7$	{$\mathrm{Agent}_3.I$ (52 %), $\mathrm{Agent}_2.P$ (73 %), $\mathrm{Agent}_1.C$, $\mathrm{Agent}_4.C$}	{$\mathrm{AGENT}_3.M$ (62 %), $\mathrm{AGENT}_2.T$ (84 %), $\mathrm{AGENT}_1.O$, $\mathrm{AGENT}_4.O$}	
任务 2	$4 \in 4 \sim 7$	{$\mathrm{Agent}_3.P$ (76 %), $\mathrm{Agent}_2.C$, $\mathrm{Agent}_1.P$ (75 %), $\mathrm{Agent}_4.P$ (78 %)}	{$\mathrm{AGENT}_3.T$ (84 %), $\mathrm{AGENT}_5.O$, $\mathrm{AGENT}_6.T$ (86 %), $\mathrm{AGENT}_7.T$ (85 %)}	
任务 3	$4 \in 4 \sim 7$	{$\mathrm{Agent}_5.P$ (76 %), $\mathrm{Agent}_7.P$ (75 %), $\mathrm{Agent}_9.P$ (78 %)}	{$\mathrm{AGENT}_1.V$ (93 %), $\mathrm{AGENT}_3.T$ (84 %), AGENT2.O, $\mathrm{AGENT}_4.T$ (85 %)}	RPA_{CTI}Agent 方案连续出现两个 “, ” 说明第 2 项约束 (子任务) 无接口可以匹配
任务 4	$4 \in 4 \sim 7$	{$\mathrm{Agent}_1.P$ (75 %), $\mathrm{Agent}_2.C$, $\mathrm{Agent}_3.P$ (76 %), $\mathrm{Agent}_4.P$ (78 %)}	{$\mathrm{AGENT}_1.T$ (86 %), $\mathrm{AGENT}_2.O$, $\mathrm{AGENT}_3.T$ (84 %), $\mathrm{AGENT}_4.T$ (85 %)}	
任务 5	$4 \in 4 \sim 7$	{$\mathrm{Agent}_1.C$, $\mathrm{Agent}_2.P$ (73 %), $\mathrm{Agent}_3.I$ (52 %), $\mathrm{Agent}_4.C$}	{$\mathrm{AGENT}_1.O$, $\mathrm{AGENT}_2.T$ (84 %), $\mathrm{AGENT}_3.M$ (62 %), $\mathrm{AGENT}_4.O$}	
⋮	⋮	⋮	⋮	⋮

子任务 3 和 4 分别有 AGENT_1 的普通接口和 AGENT_4 的普通接口完成.

在表3中, CIT 是约束项数目. 表3中不难得到, CRQA_{OVTM} Agent 的协作系统比 RPA_{CTI} Agent 的系统协同制造装配的最后成功率更高. 当增大约束项数目的 (任务更复杂) 时候, 在两者的成功率都下降的情况下, CRQA_{OVTM} Agent 系统下降率比较缓慢, 表明处理的任务越复杂, CRQA_{OVTM} Agent 系统比 RPA_{CTI} Agent 系统更有可靠的性能.

表 3　两类系统实验数据

Table 3　Experiment data of the two kinds of systems

	4 ～ 7	10 ～ 12	22 ～ 25	30～32
RPA_{CTI} Agent 系统	91.4 %	72.8 %	62.3 %	40.6 %
CRQA_{OVTM} Agent 系统	97.8 %	90.2 %	77.7 %	68.5 %

4 结论

本文的 CRQA$_{OVTM}$ Agent 模型提出了解决多 Agent 系统协作模式松耦合度不够、可靠性不高的方法, 同时也应对了协同制造装配的 "结构失配" 和 "工艺革新" 特性, 该模型使得 Agent 协同下的制造装配效率得到有效的提高. CRQA$_{OVTM}$ Agent 模型采用的 $OVTM$ 协作接口可以很好地支持普通二元协作、转换二元协作、适配性二元协作和调制性二元协作关系, 并在参数松耦合的协作下建立合同约束的虚拟组织, 由于受合同约束, 使得协同变得可信. CRQA$_{OVTM}$ Agent 支持的协同制造装配系统比以前模型更能提高开放式环境下协同制造装配的松耦合度, 也提高了可靠性.

进一步的研究方向是将研究成果渗透进目前国内外研究热点的服务计算、云计算、云制造等学科和工程领域, 推进这些领域的发展.

References

1 Barenji RV, Barenji A V, Hashemipour M. A Multi- Agent RFID-enabled distributed control system for a flexible manufacturing shop. *International Journal of Advanced Manufacturing Technology*, 2014, 71(9−12): 1773−1791

2 Liu Ming-Zhou, Wu Kun, Ma Jing, Wang Qiang, Ling Lin. A assembly process operation management method of mechanical products based on internet of manufacturing things. *China Mechanical Engineering*, 2015, **26**(16): 2183−2190
(刘明周, 吴坤, 马靖, 王强, 凌琳. 一种制造物联网环境下的机械产品装配过程运行管理方法. 中国机械工程, 2015, **26**(16): 2183−2190)

3 Grosz B J, Kraus S. Collaborative plans for complex group action. *Artificial Intelligence*, 1996, **86**(2): 269−357

4 Luo Xiao-Chuan, Liu Xing-Gang, Li Dan-Cheng, Qu Rong-Xia. Dynamic scheduling algorithm of service windows in a distributed measurement system. *Acta Automatica Sinica*, 2008, **34**(6): 690−696
(罗小川, 刘兴刚, 李丹程, 曲蓉霞. 分布式测量系统服务窗口动态调度方法研究. 自动化学报, 2008, **34**(6): 690−696)

5 Kang Ling, Wu Hua, Wang Shi-Long, Zhou Jie. Research on Service-oriented system architecture and mechanism for cloud manufacturing. *Journal of Chongqing University*, 2013, **36**(11): 66−73
(康玲, 吴华, 王时龙, 周杰. 面向服务的云制造系统架构分析. 重庆大学学报, 2013, **36**(11): 66−73)

6 Pinho D, Vivacqua A S, Medeiros S, de Souza J M. Similarity-based knowledge agents for cooperative design. In: Proceedings of the 9th International Conference on Computer Supported Cooperative Work in Design. Coventry, UK: IEEE, 2005. 417−422

7 Zuo Xing-Quan, Wang Chun-Lu, Zhao Xin-Chao. Combining multi-objective immune algorithm and linear programming for double row layout problem. *Acta Automatica Sinica*, 2015, **41**(3): 528−540
(左兴权, 王春露, 赵新超. 一种结合多目标免疫算法和线性规划的双行设备布局方法. 自动化学报, 2015, **41**(3): 528−540)

8 Lima M K, Zhang Z. A Multi-Agent system using iterative bidding mechanism to enhance manufacturing agility. *Expert Systems with Applications*, 2012, **39**(9): 8259−8273

9 Moslehi G, Mahnam M. A pareto approach to Multi-Objective flexible Job-Shop scheduling problem using particle swarm optimization and local search. *International Journal of Production Economics*, 2011, **129**(1): 14−22

10 Yang Pei-Ying, Tang Jia-Fu, Yu Yang, Pei Jin-Xiang. Minimizing carbon emissions for vehicle routing and scheduling in picking up and delivering customers to airport service. *Acta Automatica Sinica*, 2013, **39**(4): 424−432
(杨培颖, 唐加福, 于洋, 裴金翔. 面向最小碳排放量的接送机场服务的车辆路径与调度. 自动化学报, 2013, **39**(4): 424−432)

11 Wan Xiao-Qin, Yan Hong-Sen, Wang Zheng. Scheduling and self-reconifiguration of an aircraft engine assembly line in knowledgeable manufacturing. *Acta Automatica Sinica*, 2015, **41**(1): 136−146
(万晓琴, 严洪森, 汪峥. 知识化制造环境下航空发动机装配线调度及自重构. 自动化学报, 2015, **41**(1): 136−146)

12 Guo Hang, Gao Ji, Hu Bin, Fu Zhao-Yang. Cooperative design system supported by RPACTI agent in open environment. *Journal of Computer-Aided Design & Computer Graphics*, 2009, **21**(5): 694−699
(郭航, 高济, 胡斌, 傅朝阳. RPACTI Agent 支持的开放式环境下协同设计系统. 计算机辅助设计与图形学学报, 2009, **21**(5): 694−699)

周尤明 副教授, 高级工程师. 2012 年获得浙江大学计算机科学与技术学院博士学位. 主要研究方向为人工智能, Agent 协同, 知识管理. 本文通信作者.
E-mail: monitorstudent@126.com
(**ZHOU You-Ming** Associate professor, senior engineer. He received his Ph. D. degree from Zhejiang University in 2012. His research interest covers artificial intelligence, agent cooperation, knowledge management. Corresponding author of this paper.)

古华茂 副教授、浙江大学博士、博士后. 主要研究方向为人工智能, 仿生味觉, 仪器智能信息处.
E-mail: GHMSJQ@mail.zjgsu.edu.cn
(**GU Hua-Mao** Associate professor, Ph. D. graduated from Zhejiang University, postdoctor. His research interest covers artificial intelligence, bionic taste, information processing of instrument intelligence.)