国 防 科 技 大 学 学 报
第21卷 第3期 JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY Vol.21 No.3 1999

多 Agent 系统中 Agent 计算的能力理论*

毛新军 王怀民 陈火旺

(国防科技大学计算机系 长沙 410073)

摘 要 能力是规范 Agent 的一个重要抽象概念。本文提出了规划树概念以刻划在动态、不确定的多 Agent 系统中 Agent 如何通过对其规划进行合理、有效的组织来实现其任务、维护某些条件。基于规划树概念，本文给出了能力概念的形式化语义定义，获取和描述了它的一些重要属性。

关键词 Agent，多 Agent 系统，信念，能力

分类号 TP301

The Capability Theory of Agent in Multi-agent System

Mao Xinjun Wang Huaimin Chen Houwang

(Department of Computer Science, NUDT, Changsha, 410073)

Abstract Capability is an important abstract concept to specify agent A capability theory of agent computing is presented in this paper. We present the concept of plan tree to describe how agent organizes its plans effectively to fulfill its tasks and maintain conditions in the dynamic, non-deterministic multi-agent system. Based on the concept of plan tree, the formal semantics of the capability is defined and some important properties are obtained.

Key words agent, multi-agent system, belief, capability

1 前言

分布计算是目前计算机科学领域中的一项关键性主流技术和一种重要的计算方式和范型。分布计算系统由一组相对独立并能自主运作的计算实体组成。随着分布计算的应用领域变得日趋复杂和庞大，如何促进分布计算系统的开发已成为当前软件工程领域一项富有挑战性的研究课题。

Agent 是计算机科学领域中的一个重要概念。Agent 概念的自主性、交互性、社会性等特征为准确地刻划和研究分布计算系统中的计算实体提供了合理的概念模型。本文以分布计算系统的开发为背景，用 Agent 概念来刻划分布计算实体，将分布计算系统视为多 Agent 系统，提出了多 Agent 系统中 Agent 计算的能力理论以支持 Agent 计算的理论研究。

在多 Agent 系统开发过程中，能力是规范和描述 Agent 的一个重要抽象概念，原因是：(1) 在 Agent 计算过程中，理性 Agent 将根据其能力来选择规划、执行动作，因而能力是组成 Agent 体系结构的一个重要部件，是理解和分析 Agent 计算，形式化 Agent 计算过程的一个重要概念；(2) Agent 的能力是 Agent 成功地实现其任务、维护某些条件的一个重要因素；(3) 能力概念可以有效地规范和描述复杂系统的需求，这一抽象概念使得我们可以脱离系统的内部结构和具体实现细节来研究和分析复杂系统的行为；(4) 能力概念可以用于规范和描述 Agent 间的交互和通讯行为，帮助我们获取和规范多 Agent 系统的需求[1]。为了使能力概念能够有效地指导和推动多 Agent 系统的开发，必须严格、形式化地给出它的语义定义，获取和描述它的逻辑属性。

* 国家自然科学基金资助项目
1998年10月3日收稿
第一作者：毛新军，男，1970生，博士

2 形式化框架

形式化语言 L 是对命题分枝时序逻辑 CTL^{*} [2]的扩充。语言 L 的公式集由状态公式集 L_t 和路径公式集 L_s 二部分组成。设 Φ 是原子命题符号集合; $Const_{Ag}$ 是 Agent 符号集合; $Const_{AC}$ 是原子动作符号集合; $Const_P$ 是规划表达式符号集合: $Const_T$ 是规划树符号集合。这些集合均非空且递归可枚举。为了简化说明, 文中具有下列符号约定; (1) p, q 表示原子命题符号; (2) φ, Ψ, ν 表示公式符号; (3) i, j 表示 Agent 符号; (4) a, b 表示原子动作符号; (5) α, β 表示规划表达式符号; (6) $\bar{\omega}$ 表示规划符号, 其中≊ 是空树符号。

定义2.1 (语言 L 的语法) L 是由下列规则定义的最小封闭集合

SYN-1: $p \in L_t$;

SYN-2: 如果 $\Psi, \varphi \in L_t$, 则 $-\varphi, \Psi \wedge \varphi, Has_i(\bar{\omega}), K_i\varphi, \Delta\bar{\omega}\,\varphi, \{do_i(\bar{\omega})\}_p(\varphi, \Psi)$,

$\{do_i(\bar{\omega})\}_T(\varphi, \Psi), Can_i(\varphi, \Psi) \in L_t$;

SYN-3: 如果 $\varphi \in L_s$, 则 $A\varphi \in L_t$;

SYN-4: $L_t \subseteq L_s$;

SYN-5: 如果 $\Psi, \varphi \in L_s$, 则 $-\varphi, \Psi \wedge \varphi, \Psi U \varphi, < do_i(\alpha) > \varphi, [do_i(\alpha)]\varphi \in L_s$;

其中规划表达式定义如下:

SYN-6: 如果 $\varphi \in L_t$, 则 $confirm(\varphi), a \in Const_P$;

SYN-7: 如果 $\alpha, \beta \in Const_P$, 则 $\alpha; \beta\, \alpha\ \beta \in Const_P$;

规划树表达式定义如下:

SYN-8: ≊ , $\alpha \in Const_T$;

SYN-9: 如果 $\alpha \in Const_P$, 且 $\bar{\omega}, \cdots, \bar{\omega}_n \in Const_T$, 则 $< \alpha; \bar{\omega}, \cdots, \bar{\omega}_n > \in Const_T$

除了通常的命题连接词外, 语言 L 引进了其它算子包括: 时序算子 U; 路径算子 A; 动作算子< > 和[ ]; 知识算子 K; 规划树存在算子 Has; 子规划树存在量词 Δ, 规划算子 $\{\ \}_p$, 规划树算子 $\{\ \}_T$, 能力算子 Can。语言 L 还引进规划构造算子以构造复杂的规划表达式, 包括测试和验证算子 confirm; 顺序组合算子"; ", $\alpha; \beta$ 表示先执行规划 α 然后再执行规划 β; 不确定选择算子 " ", $\alpha\ \beta$ 表示不确定地选择规划 α 或者规划 β 执行。一个规划树表达式或者是空树, 或者是规划表达式, 或者是具有形如 $< \alpha; \bar{\omega}, -, \bar{\omega}_n >$ 的表达式。

定义2.2 语言 L 的一个模型 M 是一结构 $< T, <, U_{Ag}, U_{Ac}, U_p, U_T, \prod, \pi, Act, [\ ], B >$ 。其中 T 是时刻集; $< \subseteq T \times T$, < 是 T 上的偏序关系且满足过去线性; U_{Ag} 是 Agent 集合; U_{Ac} 是原子动作集合; U_p 是基于 U_{Ac} 的 Agent 规划集合; U_T 是基于 U_P 的 Agent 规划树集合; $\prod: U_{Ag} \rightarrow powerset(U_T)$, $\prod$ 定义了每个 Agent 的规划库; $\pi: \Phi \rightarrow powerset(T)$, π 是对原子命题的解释, $\pi(p)$ 定义了使原子公式 p 成立的时刻集; $Act: U_{Ag} \times U_{Ac} \rightarrow powerset(T \times T)$, Act 定义了原子动作的发生, $[t, t'] \in Act(i, a)$ 表示 Agent-t_i 在 $[t, t']$ 路径子区间(见下面定义) 中执行动作 a, 其中 t 是动作执行的起始时刻, t' 是终止时刻; $[\ ]$: $(Const_{Ag} \rightarrow U_{Ag}) \cup (Const_{Ac} \rightarrow U_{Ac}) \cup (Const_P \rightarrow U_P) \cup (Const_T \rightarrow U_T)$; $[\ ]$ 是对 Agent 符号、原子动作符号、规划符号和规划树符号的解释: $B: U_{Ag} \rightarrow powerset(T \times T)$, $(t, t') \in B(i)$ 是指 $Agent_i$ 在 t 时刻认为 t' 时刻是可能的, B(i) 满足自反性和传递性, 函数 B 用于定义 Agent 的知识。

T 中的每一时刻对应于世界的一个状态。世界状态由在该状态下为真的原子命题来表示。< 是 T 上的偏序关系, 它描述了时刻间的先后次序。在形式化模型中, 任意时刻的过去是线性和确定的, 它的将来可能是分枝的。整个形式化模型呈图1所示的树形结构。时刻 t 的一条路径是指始于该时刻, 由 t 的将来时刻构成的一条线性分枝, 它刻划了世界的某种特定发展轨迹。

定义2.3 设 $t \leq t'$, 则 $[t, t'] = \{t'' \mid t \leq t'' \leq t'\}$ 为一路径子区间。设 S_t 表示时刻 t 的所有路径的集合, S_Σ 是所有路径的集合。

L_t 中公式的可满足语义定义由模型 M 和时刻 t 给出。$M \models_t \varphi$ 表示模型 M 在时刻 t 满足公式 φ。L_s

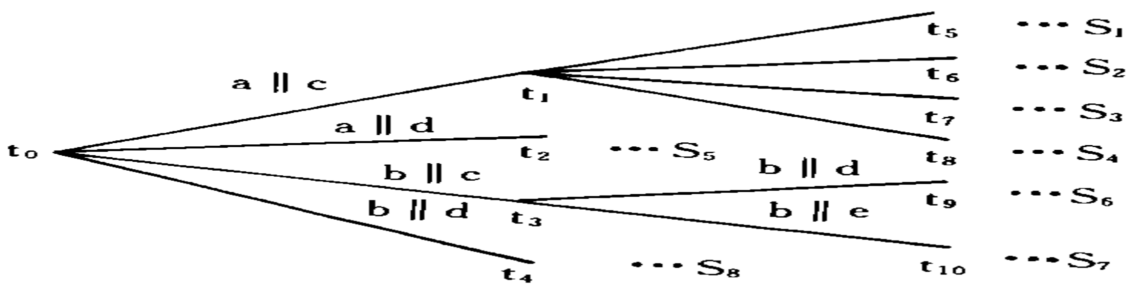

图1 形式化模型示意图

Fig. 1 The picture of the formal model

中公式的可满足语义由模型 M, 路径 S 和时刻 t 定义。$M \models_{s,t}\Psi$ 表示模型 M 在路径 S 的时刻 t 满足公式 Ψ。

定义2.4 设 $\bar{\omega}$ 为一规划树, $\Sigma_{\bar{\omega}}$ 为 $\bar{\omega}$ 的子规划树的集合, 则

如果 $\bar{\omega} = \approxeq$, 则 $\Sigma_{\bar{\omega}} = \phi$

如果 $\bar{\omega} = \alpha$, 则 $\Sigma_{\bar{\omega}} = \phi$

如果 $\bar{\omega} = <\alpha; \bar{\omega}_1, \bar{\omega}_2, \cdots, \bar{\omega}_n>$, 则 $\Sigma_{\bar{\omega}} = \{\bar{\omega}_1, \bar{\omega}_2, \cdots, \bar{\omega}_n\}$

为了简化算子 Δ 的形式化语义定义, 我们限定算子 Δ 不能嵌套且公式中的每个规划树符号只能被量化一次。设 $\varphi_{\bar{\omega},\bar{\omega}'}$ 是用 $\bar{\omega}'$ 替换 $\bar{\omega}$ 在 φ 中所有出现所得到的公式。

定义2.5 状态公式的形式化语义定义

$M \models_t p$ iff $t \in \pi(p)$

$M \models_t \Psi \wedge \varphi$ iff $M \models_t \Psi$ 且 $M \models_t \varphi$;

$M \models_t \neg\varphi$ iff $M \not\models_t \varphi$;

$M \models_t \Delta\bar{\omega}_1 \varphi$ iff $\exists\, \bar{\omega}' \in \Sigma_{\bar{\omega}_1}: M \models_t \varphi_{\bar{\omega}_1, \bar{\omega}'}$

$M \models_t A\varphi$ iff $\forall S: S \in S_t \Rightarrow M \models_{s,t} \varphi$;

$M \models_t K_i\varphi$ iff $\forall t': (t, t') \in B([i]) \Rightarrow M \models_{t'} \varphi$;

$M \models_t Has_i(\bar{\omega})$ iff $[\bar{\omega}] \in \prod([i])$

在定义路径公式的形式化语义之前, 我们首先扩充形式化模型 M 中函数 Act 的定义, 使得它能解释规划表达式。

定义2.6 函数 Act 的扩充 $Act^*: U_{Ag} \times U_P \rightarrow powerset(T \times T)$ 定义如下:

$Act^*(i, [confirm(\varphi)]) = \{[t, t] \mid M \models_t \varphi\}$

$Act^*(i, [a]) = Act(i, [a])$

$Act^*(i, [\alpha; \beta]) = \{[t, t'] \mid \exists t'': t \leq t'' \leq t'$ 且 $[t, t''] \in Act^*(i, [\alpha])$ 且 $[t'', t'] \in Act^*(i, [\beta])\}$

$Act^*(i, [\alpha \mid \beta]) = Act^*(i, [\alpha]) \cup Act^*(i, [\beta])$

定义2.7 路径公式的形式化语义定义

$M \models_{s,t} \neg\varphi$ iff $M \not\models_{s,t} \varphi$;

$M \models_{s,t} \Psi \wedge \varphi$ iff $M \models_{s,t} \Psi$ 且 $M \models_{s,t} \varphi$;

$M \models_{s,t} \Psi U \varphi$ iff $\exists t' \in S: M \models_{s,t'} \varphi$ 且 $(\forall t'': t \leq t'' \leq t' \Rightarrow M \models_{s,t''} \Psi)$;

$M \models_{s,t} < do_i(\alpha) > \varphi$ iff $\exists t' \in S: [t, t'] \in Act([i], [\alpha])$ 且 $(\exists t'': t < t'' \leq t'$ 且 $M \models_{s,t''} \varphi)$;

$M \models_{s,t} [do_i(\alpha)]\varphi$ iff $\forall t' \in S: [t, t'] \in Act([i], [\alpha]) \Rightarrow (\exists t'': t < t'' \leq t'$ 且 $M \models_{s,t''} \varphi)$;

$M \models_{s,t} \varphi$ iff $M \models_t \varphi$, 其中 $\varphi \in L_t$.

根据上述语义定义我们可以派生出其它算子。$F\varphi = trueU\varphi$, F 是“必然”时序算子。G 是 F 的对偶算子。$G\varphi = \neg F(\neg\varphi)$。A 是全称路径算子。E 是 A 的对偶算子, $E\varphi = \neg A(\neg\varphi)$。< > 和[ ]是二个动作算子以描述动作的发生以及具有的结果。直观地, 公式 $< do_i(\alpha) > \varphi$ 表示 $Agent_i$ 完成规划 α 且具有结果 φ, 公

式$[do_i(\alpha)]\varphi$表示如果 Agent$_i$ 能够完成规划 α,则具有相应的结果 φ。$<\ >$ 和$[\ ]$的语义遵循了标准动态逻辑中的定义,然而上述语义定义具有更强的灵活性,即要求 φ在规划执行过程中(而不是在规划完成之时)成立。值得注意的是,$[\ ]$不是$<\ >$ 的对偶算子。$[do_i(\alpha)]\varphi$的对偶是$-[do_i(\alpha)]-\varphi$。直觉地,$-[do_i(\alpha)]-\varphi$是指 Agent$_i$ 完成规划 α且在规划执行过程中公式 φ恒成立。$M\models_{s,t}-[do_i(\alpha)]-\varphi$ iff $\exists t'\in S:[t,t']\in Act([i],[\alpha])$ 且$(\forall t'': t<t''\le t'\Rightarrow M\models_{s,t''}\varphi)$。同理$<\ >$ 不是$[\ ]$的对偶算子。公式 $Has_i(\varpi)$表示 Agent$_i$ 具有规划树 ϖ。$K_i\varphi$表示 Agent$_i$ 具有知识 φ,或 Agent$_i$ 知道 φ。

定理2.1 知识算子 K_i 具有以下性质[1]

$\models K_i\varphi\rightarrow\varphi$

$\models K_i\varphi\rightarrow K_iK_i\varphi$

$\models K_i\varphi\wedge K_i(\varphi\rightarrow\Psi)\rightarrow K_i\Psi$

3 描述 Agent 能力的 Can 算子

将 Agent 视为一规划系统,它具有一组规划,并能对规划进行有效、合理的组织以实现其任务。将基于规划树概念来定义 Agent 能力的形式化语义。Agent 的规划树刻划了 Agent 对其规划进行组织的组织结构,封装了 Agent 在实现其任务过程中的规划选择函数。引进规划树的原因是:(1) 在多 Agent 系统中,每个 Agent 都是一个规划系统。它具有一组规划,并能基于这些规划来实现一些简单的任务和目标。然而对于一些复杂的任务而言,Agent 必须具备对其规划进行合理组织的能力。(2) 在多 Agent 系统中,Agent 的规划并不总是可执行的。Agent 的行为可能受外部条件的约束和限制,例如资源共享和冲突等因素。而多 Agent 系统是一个动态、不确定的系统。动态性体现在:多 Agent 系统中 Agent 动作执行事件和环境事件不断地发生,多 Agent 系统呈现出一种动态发展和变化的过程。不确定性表现在:多 Agent 系统中的各个 Agent 都是自主的行为实体。Agent 通过动作的执行来影响和控制系统的发展,但 Agent 的这种影响和控制是有限的,系统的发展还受其它 Agent 的动作执行事件以及环境事件的影响。因而对于多 Agent 系统中的各个 Agent 而言,系统的发展是不确定的。在任一时刻,Agent 既不能完全控制系统的发展,也不能完全准确地判断出其他 Agent 的行为以及系统发展的实际轨迹。因此 Agent 仅仅通过更为详细、更为具体的规划来实现其任务显然是不切实际的,也是不可行的。Agent 必须具备合理、有效地组织和调整其规划的能力,必须根据系统发展的实际情况来选择规划、执行动作。基于上述分析和讨论,我们认为基于规划树来刻划 Agent 对其规划进行组织的组织结构、描述 Agent 在任务实现过程中的规划选择函数是比较恰当的。规划树由称为根结点的规划以及一组子树组成。在 Agent 实现其任务、维护某些条件的过程中,Agent 将基于规划树来选择规划、执行动作。其基本思想是:Agent 选择并执行根结点的规划,并根据规划的执行情况选择某一棵子树进一步执行。一个 Agent 能够通过规划树来实现 φ且维护 Ψ 使之恒成立当且仅当它可执行根结点规划,规划的执行或者使 φ成立并维护 Ψ,或者 Agent 可进一步地选择某一棵子树来实现 φ并维护 Ψ。Agent 将根据根结点规划的执行情况来选择这一子树。Agent 如何选择子树、选择哪棵子树执行将取决于 Agent 的知识和信念。

我们首先给出算子 $\{\ \}_P$ 和 $\{\ \}_T$ 的形式化语义定义。直觉地,公式 $\{do_i(\alpha)\}_P(\varphi,\Psi)$ 表示 Agent$_i$ 知道它能通过规划 α的执行来实现 φ且维护 Ψ。

定义3.1 算子 $\{\ \}_P$ 的形式化语义定义

(1) $M\models_t\{do_i(confirm(\nu))\}_P(\varphi,\Psi)$ iff $M\models_t K_i(\Psi\wedge\nu\wedge\varphi)$

(2) $M\models_t\{do_i(a)\}_P(\varphi,\Psi)$ iff $M\models_t K_i(E<do_i(a)>true\wedge A([do_i(a)]K_i\varphi\wedge-<do_i(a)>-K_i\Psi)$

(3) $M\models_t\{do_i(\alpha;\beta)\}_P(\varphi,\Psi)$ iff $M\models_t\{do_i(\alpha)\}_P(\{do_i(\beta)\}_P(\varphi,\Psi),\Psi)$

(4) $M\models_t\{do_i(\alpha|\beta)\}_P(\varphi,\Psi)$ iff $M\models_t\{do_i(\alpha)\}_P(\varphi,\Psi)$ 或 $M\models_t\{do_i(\beta)\}_P(\varphi,\Psi)$

基于算子 $\{\ \}_P$,下面我们给出算子 $\{\ \}_T$ 的形式化语义定义。直觉地,公式 $\{do_i(\varpi)\}_T(\varphi,\Psi)$

[1] 本文定理的证明请参阅文献[3]

表示 $Agent_i$ 知道它能通过规划树 $\bar{\omega}$ 来实现 φ 同时维护 Ψ。

定义3.2 算子 $\{\quad\}_T$ 的形式化语义

(1) $M \models_t \{do_i(\approxeq)\}_T(\varphi, \Psi)$ iff $M \models_t K_i(\varphi \wedge \Psi)$

(2) $M \models_t \{do_i(\alpha)\}_T(\varphi, \Psi)$ iff $M \models_t \{do_i(\alpha)\}_P(\varphi, \Psi)$

(3) $M \models_t \{do_i(<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>)\}_T(\varphi, \Psi)$ iff

$M \models_t \{do_i(\alpha)\}_P(\Delta<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n> : \{do_i(<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>)\}_T(\varphi, \Psi), \Psi)$

(1) 指出 $Agent_i$ 在时刻 t 知道它能通过空规划树来实现 φ 并维护 Ψ 当且仅当在该时刻 $Agent_i$ 知道 φ 和 Ψ 均成立。(2) 指出 $Agent_i$ 在时刻 t 知道它能通过由单一规划 α 所构成的规划树来实现 φ 并维护 Ψ 当且仅当在该时刻 $Agent_i$ 知道它能通过规划 α 的执行来实现 φ 并维护 Ψ。(3) 指出 $Agent_i$ 在时刻 t 知道它能够通过规划树 $\bar{\omega} = <\alpha; \bar{\omega}_1, \bar{\omega}_2, \cdots, \bar{\omega}_n>$ 来实现 φ 并维护 Ψ 当且仅当在该时刻 $Agent_i$ 知道它能执行规划 α 以实现 $\Delta<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n> : \{do_i(<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>)\}_T(\varphi, \Psi)$ 并知道在执行规划 α 的过程中 Ψ 恒成立。公式 $\Delta<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n> : \{do_i(<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>)\}_T(\varphi, \Psi)$ 在时刻 t 成立当且仅当存在规划树 $<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>$ 的一棵子树 $\bar{\omega} \in \{\bar{\omega}_1, \cdots, \bar{\omega}_n\}$ 使得 $\{do_i(\bar{\omega})\}_T(\varphi, \Psi)$ 在时刻 t 成立。

引理3.1 $\models K_i(E<do_i(a)>true \wedge A([do_i(a)]K_i\varphi \wedge -<do_i(a)>-K_i\Psi)) \wedge K_i\Psi \rightarrow K_iE(K_i\Psi U K_i\varphi)$

引理3.1表明, 对于任意时刻如果 $Agent_i$ 知道原子动作 a 能够发生且知道如果该动作发生必能实现 φ 同时维持 Ψ 使之恒成立且知道 Ψ 在该时刻成立, 则在该时刻 $Agent_i$ 知道存在一条路径且公式 $(K_i\Psi U K_i\varphi)$ 在该路径上成立。

定理3.1 $\models \{do_i(\alpha)\}_P(\varphi, \Psi) \wedge K_i\Psi \rightarrow K_iE(K_i\Psi U K_i\varphi)$

定理3.1表明如果 $Agent_i$ 知道它能通过规划 α 的执行来实现 φ 并维护 Ψ 且知道 Ψ 在该时刻成立, 则它知道存在一条路径且公式 $(K_i\Psi U K_i\varphi)$ 在该路径上成立。可采用归纳法并结合引理3.1来证明该定理。

定理3.2 $\models \{do_i(\bar{\omega})\}_T(\varphi, \Psi) \wedge K_i\Psi \rightarrow K_iE(K_i\Psi U K_i\varphi)$

定理3.2表明如果 $Agent_i$ 知道它能通过规划树 $\bar{\omega}$ 来实现 φ 同时维护 Ψ 且知道 Ψ 在该时刻成立, 则它知道存在一条路径且公式 $(K_i\Psi U K_i\varphi)$ 在该路径上成立。可采用归纳法并结合定理3.2来证明该定理。

下面我们给出 Can 算子的形式化语义定义。公式 $(Can_i(\varphi, \Psi)$ 表示 $Agent_i$ 能够实现 φ 同时维护 Ψ 使之恒成立。Agent 在某一时刻的能力不仅与 Agent 的规划以及对规划的组织能力(即 Agent 具有的规划树)相关, 而且还与在该时刻规划树的执行情况以及 Agent 的知识相关联。形式化地, Can 算子具有下列语义定义。

定义3.3 算子 $Can_i(\varphi, \Psi)$ 的形式化语义

$M \models Can_i(\varphi, \Psi)$ iff $M \models K_iHas_i(\bar{\omega}) \wedge \{do_i(\bar{\omega})\}_T(\varphi, \Psi)$

根据上述语义定义, 公式 $Can_i(\varphi, \Psi)$ 在时刻 t 成立当且仅当在该时刻 $Agent_i$ 知道它具有某棵规划树且它能通过该规划树来实现 φ 并维护 Ψ。下面我们给出算子 Can 的一组公理并证明它们是可靠的。

公理3.1 $K_iHas_i(\approxeq) \wedge K_i(\varphi \wedge \Psi) \rightarrow Can_i(\varphi, \Psi)$

公理3.2 $K_iHas_i(\alpha) \wedge \{do_i(\alpha)\}_P(\varphi, \Psi) \rightarrow Can_i(\varphi, \Psi)$

公理3.3 $K_iHas_i(<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>) \wedge \{do_i(\alpha)\}_P(\Delta<\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n> : \{do_i(\alpha; \bar{\omega}_1, \cdots, \bar{\omega}_n>)\}_T(\varphi, \Psi), \Psi) \rightarrow Can_i(\varphi, \Psi)$

定理3.3 公理3.1～公理3.3是可靠的, 可根据 Can 算子的语义定义来证明该定理。

4 Can 算子的逻辑属性

算子 Can 在知识和规划层次上刻划了 Agent 的能力。公式 $Can_i(\varphi, \Psi)$ 在时刻 t 成立当且仅当在该时刻 $Agent_i$ 知道它具有某棵规划树且它能以该规划树作为其实现任务的规划选择函数来实现 φ 同时维护 Ψ 使之恒成立。基于上述定义, 如果 $Agent_i$ 能够实现 φ 同时维护 Ψ 且它知道 Ψ 在当前时刻成立, 则 $Agent_i$ 知道存在一条路径, 公式 $K_i\Psi U K_i\varphi$ 在该路径上成立。形式化地, 我们具有下列定理。

定理4.1 $\models Can_i(\varphi, \Psi) \wedge K_i\Psi \rightarrow K_iE(K_i\Psi U K_i\varphi)$。可根据 Can 算子的定义以及定理3.2来证明。

定理4.2 $\models Can_i(\varphi, \Psi) \rightarrow K_iCan_i(\varphi, \Psi)$

定理4.2表明如果 Agent 能够实现φ同时维护 Ψ,则 Agent 知道它能实现φ同时维护 Ψ。可根据 Can 算子的语义定义以及定理2.1来证明该定理。

定理4.3 $\models -Can_i(false, \Psi)$。定理4.3指出任何 Agent 都不能够实现 false。可采用反证法来证明。

定理4.4 $\models -Can_i(\varphi, false)$。

该定理指出任何 Agent 都不能够维护 false 使之恒成立。可采用反证法来证明。

5 结论

在多 Agent 系统开发过程中,能力是规范和描述 Agent 的一个重要抽象概念。本章提出了多 Agent 系统中 Agent 计算的能力理论以支持 Agent 计算的理论研究。我们基于规划树概念给出了能力概念的形式化语义定义。规划树是对 Agent 的规划组织结构以及在实现任务过程中 Agent 的规划选择函数的一种抽象表示。在 Agent 实现其任务、维护某些条件的过程中, Agent 将基于规划树来选择规划、执行动作。Agent 某一时刻的能力不仅与 Agent 的规划以及对规划的组织能力(即 Agent 具有的规划树) 相关,而且还与该时刻规划树的执行情况以及 Agent 的知识相关联。Agent 的知识和信念将影响 Agent 的规划选择。本文给出了描述 Agent 能力的 Can 算子的语义定义,获取并描述了 Can 算子的一些重要的逻辑属性。

参考文献

1 Singh M P. Multi-agent System: A Theoretical Framework for Intentions, Know-how, and Communications. Springer-Verlag, 1994

2 Emerson E A. Temporal and Modal Logic. Handbook of Theoretical Computer Science. 1990

3 毛新军. Multi-agent 系统中 Agent 计算的理论框架. 博士学位论文[国防科技大学研究生院], 1998