

Agent 系统

推理模型

知识库

(5)

13-15, 41

组织结构设计 计算机科学 1999Vpl. 26No. 11

信息处理

Agent 系统的推理模型与知识库组织结构的研究^{*}

A Design of Inference Model and Knowledge Base Architecture in the Processor of Agent System

夏幼明 徐天伟 张春霞 TP18

(云南师范大学计算机科学系 昆明 650092)

Abstract This paper analyzes the organization structure and the working pattern of inference mechanism in the processor of Agent. Based on the former research of Agent system, the authors study the inference model of the processor, and submit the controlling mechanism and the coordinating inference mechanism about inference model. Moreover, we design an organization structure and an inference scheme of knowledge base.

Keywords Processor of Agent, Inference model, Knowledge base, Structure

1. 概述

Agent 思想出现于 70 年代, 80 年代后期才逐渐发展起来, 现已融入到计算机的各个应用领域。Agent 的思想最初来源于社会学、经济学与生态学等学科。美国 Stanford 大学的 Y. Shoham 教授提出的 AOP 理论模型中^[1], 将 Agent 定义为包含了诸如信念、承诺、能力和决定等精神状态的实体, 并利用基于显示时间点逻辑语言对精神状态进行了形式化定义。1994 年, Minsky 在《Communication of ACM》杂志上对 Agent 这个概念做了较明确的说明: “Agent 是一些具有特别技能的个体”, 对计算机系统而言, Agent 是指“当你试图说明完成一些任务的机器而无需了解其如何工作时, 即将其处理为黑箱时, 就称其为 Agent”。F. Bacchus 等人在 1986 年提出一种方法——“随机世界方法”^[2], 此方法根据统计知识库确定 Agent 知识的可信度, 在执行的过程中采用了有向推理和非单调推理等策略。本文作者在原有的 Agent 系统研究基础上^[1], 对 Agent 系统的处理器中的推理模型进行了研究, 本文基于不精确推理和反向推理线索化等方法给出推理模型的控制、协调推理机制, 并提出一种知识库的组织结构和控制推理算法。文中对知识(规则)的可信度表示法提出一个形式化的描述, 分析已有的基于 Agent 的规则的可信度评测策略, 并做了适当的改进, 在算法设计的过程中, 主要到考虑算法的可实现性和实用性。

2. Agent 的信息处理器

Agent 是一个具有信息处理能力的主动实体^[1], 具有与外界交互的感知器、通讯机制, 对信息进行存储加工的信息处理器、记忆库, 同时还应该具有根据共同目标及自己的职责所产生的目标模块以及反作用于外部环境的效应器。

Agent 的信息处理器是体现 Agent 智能行为的最重要的部件。Agent 的信息处理器由信号/信息过滤器、控制器、符号推理机制、类比匹配机制、内部执行机制及知识库组成。其中, 知识库中包含两类知识, 一类是规则, 另一类是知识块。信息处理器的组织结构见图 1。信息处理器在接收到信号/信息后, 先对其进行过滤、抽象、聚合, 使其形成可以与客观世界的对象联系起来的有意义的符号, 然后由类比匹配机制将这些符号及特征与知识库中的知识块进行模糊匹配。如果能查找到高度匹配的知识块, 相应的知识块被用来处理信息并产生决策; 如果只能部分匹配, 则控制器驱动内部执行器将被匹配的部分知识作为符号, 然后运用推理机制及知识库中的规则处理信息。

3. Agent 推理模型

体现 Agent 智能行为的关键环节是信息处理器中推理部分的组织结构和控制推理机制。我们将类比机制、控制推理机制和知识库结合在一起, 组成一推理子系统; 并提出一个问题求解的推理模型^[3]。此模型的

^{*} 本课题得到云南省自然科学基金资助, 项目编号: 97F018M, 并得到中科院陆汝钊研究员承担的国家“九五”攻关项目的资助, 项目编号: 96-729-01-06-09。夏幼明 副教授, 工学硕士, 从事人工智能、数据库等领域研究。

组成部分为:求解控制器、问题分析器、问题分解机制、触发器和 Agent 求解库。Agent 的推理模型及其系统的组织结构如图 2 所示。

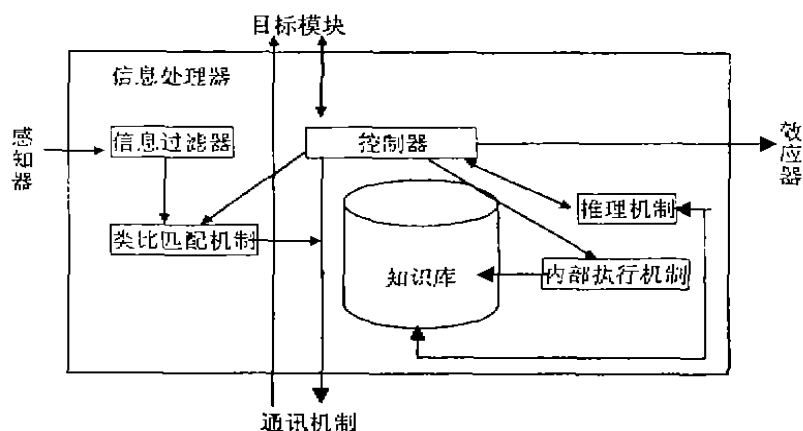


图 1 信息处理器结构

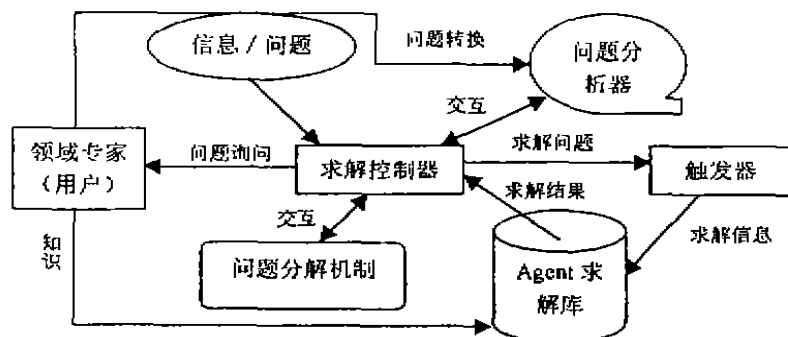


图 2 Agent 系统的推理模型的组织结构

问题求解推理模型首先由求解控制器将要处理的信息转换成问题表示,并通过对问题的分析(借助于问题分析器)将问题分解为若干子问题(借助于问题分解机制),在 Agent 求解知识库的支持下问题进行求解。触发器由规则触发器、事实触发器和知识块触发器组成,用于对 Agent 求解库的消息传送,驱动推理过程。

触发器中事件驱动的规则^[3]有如下形式:

```

RULE(规则名)[(参数),...]
WHEN(事件表达式)
IF(条件 1)THEN(动作 1)
.....
IF(条件 i)THEN(动作 i)
END-RULE[(规则名)]

```

〈条件 i〉($i=1,2,\dots,n$)是某种逻辑中的任意一个合法的逻辑公式。〈动作 i〉($i=1,2,\dots,n$)既可以是系统预先定义的一些标准动作,也可以是用户定义的

一个动作,或是用某种语言编写的一个程序。

每个事件都有一个事件名标识,并有开始(发生)时间 $B(e)$ 、终止(发生)时间 $E(e)$ 和发生期 $D(e)$ 等属性,其中 e 是一个事件名。时间可分为如下几类:

- (1)与时间相关的基本事件;
- (2)与状态相关的基本事件;
- (3)与信号有关的基本事件;
- (4)与公共变量有关的基本事件;
- (5)与通讯环境有关的事件。

事件之间存在如下运算:(1)同时发生运算;(2)选择发生运算;(3)合并发生运算;(4)相继发生运算;(5)之前发生运算;(6)之后发生运算;(7)不发生运算。

问题分析器由知识规则和经验规则组成,确定问题归属的领域及分类。经验规则由不精确推理机制实现。问题分解机制负责对问题进行划分,问题划分是按照组成问题的逻辑关系,比如说与(and)、或(or)等作

为划分的基础,将原问题进行转换或划分为若干子问题,形成树结构,子问题之间的求解关系类似于在博弈中的求极小和极大的算法(α - β 算法);另一方面,利用知识库的规则进行后推理(与规则的结论匹配)产生子问题,将原问题分解。

首先,对某一求解的问题要确定其归属的领域,在此模型中的领域专家就是指解决这类问题的专家,求解控制器与领域专家之间的协调关系表示在求解问题过程中,当整个推理过程发生障碍无法继续进行的时候,由领域专家参与提供求解的辅助信息,其中包括将问题转换成一个等价的问题,或向系统提供更多的推理知识和领域知识。这与专家系统的全自动问题求解过程有本质的差异^[6]。

4. 问题求解过程

求解问题的整个过程由求解问题控制器完成。在推理过程中利用不精确推理机制,在下面的算法中 $C(P)$ 表示公式 P 的可信度,其算法 S 如下:

S1 由前提 P 的不确定性,给出:

①前提 P 为真时结论 Q 的可信度;②前提 P 为假时结论 Q 的可信度;③前提 P 不确定时结论 Q 的可信度。即,

$$R_1: P \rightarrow (Q \text{ and } C(Q) = n_1);$$

$$R_2: \neg P \rightarrow (Q \text{ and } C(Q) = n_2);$$

$$R_3: (P \text{ and } C(P) = n_3) \rightarrow (Q \text{ and } C(Q) = n_4);$$

其中,括号部分是说明公式 P (前提或结论)或规则 R 的可信度。

S2 由规则的不确定性,给出:

①若前提 P 为真则结论 Q 为真时的可信度;②若前提 P 为真则结论 Q 为假时的可信度;③前提 P 对结论 Q 不存在不确定性影响时规则 R 的可信度。即,

$$R_1: (P \text{ and } C(R_1) = n_1) \rightarrow (Q \text{ and } C(Q) = n_2);$$

$$R_2: (P \text{ and } C(R_2) = n_3) \rightarrow \neg Q \text{ and } C(\neg Q) = n_4);$$

$$R_3: P \rightarrow (Q \text{ and } C(R_3) = n_5);$$

S3 根据规则 R 中前提 P 的可信度和规则 R 的可信度,求出结论 Q 的可信度。即,

$$R: (P \text{ and } C(P) = n_1 \text{ and } C(R) = n_2) \rightarrow (Q \text{ and } C(Q) = n_3);$$

S4 根据分别由两个独立的前提 P_1, P_2 , 得到结论 Q 的可信度,求出这两个前提的组合所导致的结论 Q 的可信度。即,

$$\text{若 } R_1: P_1 \rightarrow (Q \text{ and } C(Q) = n_1),$$

$$R_2: P_2 \rightarrow (Q \text{ and } C(Q) = n_2); \text{ 则}$$

$$R_3: P_1 \theta P_2 \rightarrow (Q \text{ and } C(Q) = n_3); \text{ 其中, } \theta \text{ 表示一种逻辑运算, } C(Q) \text{ 的值与该运算有关。}$$

S5 根据两个前提 P_1, P_2 的可信度,求出这两个

前提合取的可信度。即,

$$R: ((P_1 \text{ and } C(P_1) = n_1) \text{ and } (P_2 \text{ and } C(P_2) = n_2)) \rightarrow (P_1 \text{ and } P_2 \text{ and } C(P_1 \text{ and } P_2) = \min\{n_1, n_2\});$$

S6 根据两个前提的可信度,求出这两个前提析取的可信度。即,

$$R: ((P_1 \text{ and } C(P_1) = n_1) \text{ and } (P_2 \text{ and } C(P_2) = n_2)) \rightarrow ((P_1 \text{ or } P_2) \text{ and } C(P_1 \text{ or } P_2) = \max\{n_1, n_2\});$$

在不精确推理的机制中,所涉及的前提和结论分别对应规则中的前提和结论。规则的不确定性(可信度)及推理结果的不确定性的知识由语义框架数据字典提供,这将涉及相应领域专家的知识,相关的研究将另文介绍。

由于在推理过程中采用将问题划分为子问题进行求解的方法,所以在控制器中建立一个问题的存储栈,用于存放在求解过程中产生的子问题。存储栈初始为原问题,当栈为空时求解结束。

Agent 求解问题库存放规则和知识块,知识块中存放已有的解决此问题的知识,解决问题的方案如下:

1. 将问题送到问题分析器,确定启用 Agent 求解问题库的规则或直接调用 Agent 求解问题库的知识块(知识块可视为规则链,即解题路径)。若没有直接可利用的规则或解决问题的知识块,则转到 3。

2. 若是规则求解,利用推理机制及库中的规则处理信息,形成新的知识块;如果没有可用知识块,则 Agent 求解问题库中的规则被用来处理信息。Agent 求解问题库中的规则及知识块随着问题的处理不断被添加及更新。若在推理的过程中 Agent 求解问题库中没有规则及知识块直接解答问题则将当前启用的规则的前提转化成新问题(子问题),转到 3;否则,正在求解的问题得到解决,转到 5。

3. 对得到的问题利用问题分解算法对问题进行划分,转换成为若干子问题进行求解。若划分成功,则依次将子问题送到问题控制器,回到 1;否则,转到 4。

4. 咨询领域专家,根据从领域专家处得到的信息,决定结束(问题无解),或者将原问题转换为另一种形式的等价表示,或者将专家所提供新的知识存到求解库中,并且转到 1。

5. 若求解控制器中的问题栈为空,则结束(问题得到解答);否则取出问题,求解控制器采用并行处理机制,对不关联的问题进行并行推理处理,以提高求解问题的效率。转到 1。

在上述算法中提到的知识库更新等操作,这涉及到知识库的维护,此项维护工作包括对知识库进行完整性、冗余性和相容性的检查与修改等工作,系统将采用非单调推理机制等技术进行知识库的管理,有关研究将在另文介绍。

(下转第 41 页)

它只支持微软各种操作系统下的 Web 服务器,它们的最低版本是:

- Windows NT Server 4.0: Microsoft IIS (Internet Information Server) 3.0;
- Windows NT Workstation 4.0: Microsoft Peer Web Services 3.0;
- Windows 95: Microsoft PWS (Personal Web Server) 1.0, 中文 Win95 应配中文的 PWS。

ASP 的安装分为 Windows NT 和 Windows95 两种情况:

1) 对于 NT, IIS 及 ActiveX Server Pages 都包含在 Service Pack 3 中;

2) 对于 Win95, 可使用 Visual Studio 服务器构件下的 ActiveX Server Pages 安装选项, 安装成功后, 在程序管理器中会找到一个 ActiveX Server Pages Roadmap 主页。

4. ASP 方式访问数据库实例

一个发布 SQL Server 库 Readerbase (DSN 为 ReaderSource) 中 reader 表的 (displayreader.asp) 完整的 ASP 脚本代码: (其中符号 "<%" 和 "%>" 是 ASP 脚本的专用定界符)

```
<html>
<head>
<title>Readers Listing </title>
</head>
<body>
<p>Readers Listing </font>
<% Set Connection=Server.CreateObject
```

```
("ADODB.Connection")
Connection.Open "ReaderSource"
Set RS=Connection.Execute("SELECT * FROM reader")
<%
>/p>
<table border=1>
<TR>
<% for i=0 to RS.Fields.Count-1%>
<TD><%=RS(i).Name%></TD>
<% Next %>
</TR>
<% Do while Not RS.EOF %>
<TR>
<% for i=0 to RS.Fields.Count-1%>
<TD><%=RS(i)%></TD>
<% Next %>
</TR>
<% RS.MoveNext
Loop
RS.Close
Connection.Close
%>
</table>
</body>
</html>
```

随着网络技术的不断发展, 浏览 Web 页将成为我们获取信息资源的重要手段, 通过网络访问数据库使我们所获取信息资源的范围大大增加。笔者坚信, 以浏览器方式访问数据库的实践必将具有深远影响。

参考文献

- 1 Desborough J. Intranet Web Development. Prentice Hall, 1996
- 2 孙燕唐, 等. 面向 Windows 的 Internet 网络应用与开发. 北京: 电子工业出版社, 1997

(上接第 15 页)

5. Agent 求解库的组织策略

在 Agent 求解库 (知识库) 中, 规则的组织结构直接影响到推理的效率。因此, 我们按规则的存储方式将其分为两类, 其结构都采用哈夫曼树结构对规则进行编码排序。一种为领域专家指导下的领域知识库, 另一种为基于应用的加权知识库。

1. 领域知识库: 由领域专家提供组成规则基的各个规则使用频率, 我们视为“权”。按照规则的权值的大小构造相应的规则的哈夫曼树。

2. 加权知识库: 初始时, 将所有规则的权都规定为相同的, 构成相应的哈夫曼树。然后, 在应用当中, 根据规则的使用频率对规则的权进行修改; 根据修改的结果对规则哈夫曼树做相应的改动。

3. 在两个知识库的哈夫曼树的结点 (规则) 都有推理线索指针, 用于推理过程中推理线索化, 从推理的过程中做线索化工作。例:

规则①: $p \rightarrow q$, 规则②: $q \rightarrow s$; 其中, p, q, s 是逻辑表达式。

在线索化过程中, 规则②的前驱线索指向规则①, 规则①的后继线索指向规则②。

4. 在知识库中, 可能出现由多个规则得到同一结论的情况, 比如: 规则⑦: $p1 \rightarrow q1$, 规则⑧: $p2 \rightarrow q1$, 规则⑨: $p3 \rightarrow q1$ 。在此情形下, 线索结点采用兄弟线索的结构加以构造, 用于解决此问题。

参考文献

- 1 Shoham Y. Agent oriented programming. Artificial Intelligence, 1993, 60(1)
- 2 Bacchus F et al. From statistical knowledge bases to degrees of belief. Artificial Intelligence, 1996, 87
- 3 夏幼明, 等. 软件 Agent 初步研究. 云南师范大学学报, 1999, 19(3)
- 4 刘弘, 等. 软件 Agent 的构筑. 计算机科学, 1998, 25(2)
- 5 李绍原, 等. 主动数据库. 见: 数据库技术最新进展, 清华大学出版社出版
- 6 吴建林, 等. 专家系统与多 Agent 协作系统. 计算机科学, 1998, 25(4)