Rete匹配算法在知识库机中的实现方案*

郭福顺 张学海 程退安 廖明宏
(哈尔滨工业大学计算机系，150006)

A SCHEME OF REALIZING THE RETE MATCHING ALGORITHM IN KNOWLEDGE－BASED MACHINE

Guo Fushun, Zhang Xuehai, Chen Tuian and Liao Minghong
(*Department of Computer Science, Harbin Institute of Technology*, 150006)

Abstract At present, the Rete algorithm is considered as one of the most fast algorithms for the production systems. Based on the main character of Rete, this paper presents a scheme of realizing it in knowledge－based machine, and discusses the technique problems such as pipeline match in the scheme. The aim is to enhance matching efficience and to improve process of production systems.

Keywords Production systems, Rete algorithm , knowledge－based machine, pipeline match

摘要 Rete算法是目前公认的用于产生式系统的高速匹配算法。本文根据Rete算法的主要特点，提出在知识库机中实现Rete算法的方案，并讨论了该方案中的流水匹配等技术问题，目的在于提高匹配效率，从而加快产生式系统的执行速度。

关键词 产生式系统，Rete算法，知识库机，流水匹配

一、引 言

产生式系统（简称PS）已被广泛地运用于人工智能的各个领域中，特别是作为专家系统开发工具，已取得了很大的成果。例如，著名的专家系统R_1就是用产生式系统程序设计语言OPS_5来实现的。PS的优点是它便于知识的表示和编码，因而是专家系统经常采用的知识表达方法，其缺点是运行效率较低（在单机环境下尤为明显），这一方面是由于PS自身要消耗90%的时间进行模式匹配的特点所决定的，另一方面也暴露了传统的Von Neumann机难于适应非过程性语言的弱点。为提高PS的运行效率，Forgy曾提出过著名的Rete匹配算法，该算法是一个对多模式集合和多目标集合进行比较的有效方法。它能找到与每一个模式进行匹配的全部目标。然而，由于受限于现有的硬件环境，目前Rete算法尚不能发挥它应有的高速匹配

本文1991年11月21日收到。*本课题是“863”研究项目。

效率。

这里，我们提出一种基于知识库机实现Rete匹配算法的方案，该方案有以下几个主要特点：

1）为方便用户，系统将在OPS_5语言基础上用知识库操作语言 KBL 扩展知识库的建立、管理、维护和使用等功能；

2）以 Rete 算法为理论依据，用后端智能存储器实现高效流水匹配操作；

3）系统以数据流驱动方式工作，信息按包（数据元素包、产生式包、示例包、命令包等）的方式在系统内流动，系统网络中各处理结点根据包的具体内容（格式）进行相应的处理；

4）前后端机采用双向通讯，系统负载平衡和通讯开销较小。

二、PS 及 Rete 算法

从计算模型角度来讲，PS 及 Rete 算法实际上是两种既有其特殊之处又有其共同特征的计算模型。

产生式的概念最早可溯源于巴甫洛夫的条件反射学说，把刺激（相当于条件）、反应（相当于动作）作为高级神经活动最基本的成份。1943年 Post 提出产生式的形式理论，并证明其计算能力与 Turing 机是等 价的。后来 Chomsky 建立形式语言理论时，第一次引进产生式作为语法规则的描述，但其产生式比较简单，离应用尚有距离。直到1972年，Simon 和 Newell 在认知心理的信息加工理论发展基础上（问题求解、逻辑推理和自然语言理解等高级思维都是采用产生式形式）把原来简单形式的产生式作了扩充。左部扩充为复杂的模式，可带有变量，可为若干谓词 的合取，可与一大批工作单元的内容相匹配。这种模式可以是外来的信息，也可以是内部心理形式的信息。而产生式右部动作扩充为可以是一整段顺序执行的程序。作了这样扩充之后，产生式系统便成为当今流行的以产生式规则表达知识的知识库系统，它具体地由规则库、数据库及推理机三部分组成（见图1）。其中，推理机可以看作一个有三个动作状态的循环，即匹配规则（match－rules）、选取规则（select－rules）及执行规则（execute－rules）的有限状态机（finite－state machine），它根据数据库中的事实，利用规则库中的产生式规则进行推理。规则库和数据库则构成知识库。

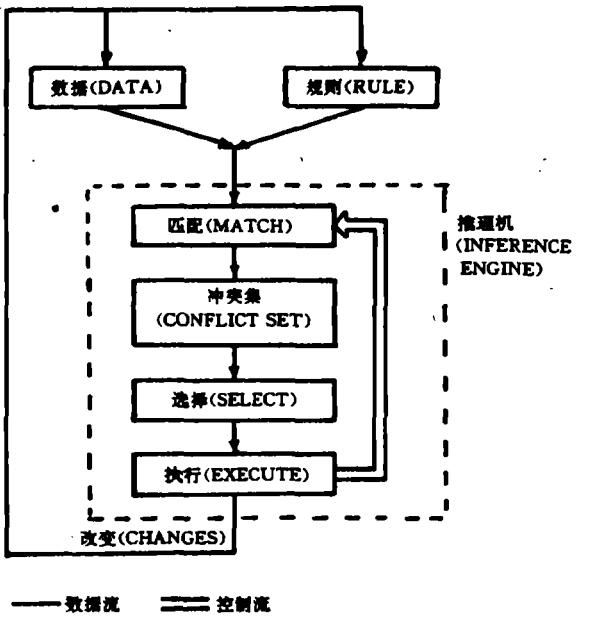

图1 产生式系统结构

一般说来，PS 具有以下几个优点：

1）if－then 形式与人类思维方式接近，适合于表示人类的经验性知识；

2）每条规则在结构上和功能上具有相对的独立性，其表现形式也符合一致性；

3）书写规则简便、易读，便于修改和补充知识；

4）高度模块化。

另一方面，PS 系统中各产生式的相对独立性，使系统运行时的数据流动和控制流动难于掌握，甚至有不终止的可能；况且，PS 系统在匹配阶段开销很大，影响了运行效率。针对 PS 的这些缺点，Forgy 设计了一种多模式与多客体之间的叫 Rete 的匹配算法，Rete 算法可普遍用于 PS 的解析器中，并被公认为目前用于 PS 的高速匹配算法之一。

在 Rete 算法中，PS 系统经过编译而生成 Rete 网。Rete 网由根节点、单 输入节点、双输入节点和终止节点构成。Rete 网和工作存储器 WM、冲突归结等一起配合工作。在匹配过程中，数据的流动以点火的方式从网的上端按一定规律向下端的节点流动，因此，数据和控制的流动较易掌握。

典型的 OPS_5 解析器便是采用 Rete 算法来完成匹配的。在 OPS_5 解析器中，匹配过程的输出和冲突归结（图1中的选择部分）的输入是一个叫做冲突集的集合，冲突集是一个有序对的集合，形如：

（产生式　与产生式左手部匹配的元素表）

该有序对又叫示例集，Rete 匹配算法就是一个计算冲突集的算法，即将产生式的左手部集合与 WM 元素集合进行比较，并从中找出全部示例。Rete 算法具有如下两个重要特征：

1）用空间换取时间，消除每个运行周期里重复的匹配测试；

2）利用产生式内部及产生式之间的结构相似性，共享匹配测试结果。

```
(P Plus                          (P Time
  (Goal ↑Type Simplify             (Goal ↑Type Simplify
        ↑Object<n>)                      ↑Object<x>)
  (Expression ↑Name<n>             (Expression ↑Name<x>
              ↑Arg1O                           ↑Arg1O
              ↑Op +                            ↑Op *
              ↑Arg3<x> )                       ↑Agrz2<x> )
              →……)                             →……)
```

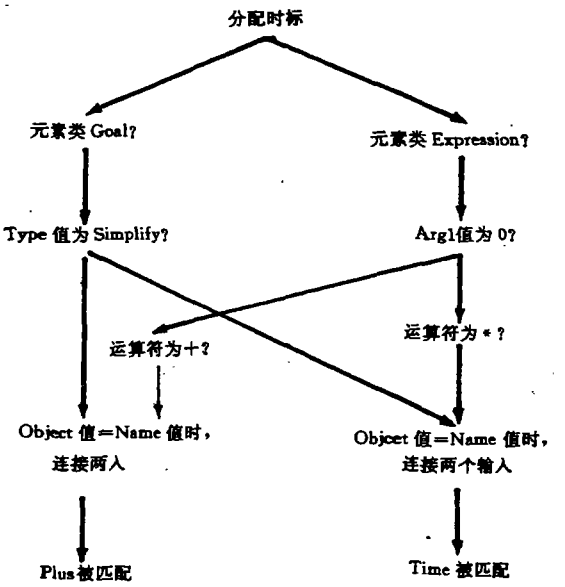

图2　Plus 和 Time 结成的 Rete 网

因此，Rete 算法不必反复对集合进行操作便能有效地对多模式与多客体进行匹配。图2是

用 OPS_5 书写的两条规则 Plus 和 Time 及其相应的 Rete 网。其中，P 为标识符；Plus 和 Time 为规则名；符号→的左手部为条件元素，每一个条件元素包括一类名（如 goal）后跟若干属性和属性值，全部用括弧括起来；而属性则用前置操作符↑来区别；符号→的右手部是动作部分（因与 Rete 网无关，故在图2中被省略），它说明如果规则 Plus 和 Time 匹配成功（即有 WM 元素同时匹配两条规则的条件元素），则其右手部点火，同时有示例产生并被包括在冲突集中，而所有出现的常量（如 Simplify）被匹配且变量<n>和<x>在 WM 元素中均有标识值。

在图2中，网络的根节点（分配时标）是模式匹配器的输入端，当根节点接收到被发往模式匹配器的标记以后。便将标记复制给它的全部后继节点。根节点的后继节点对内部元素特征（属性和常量值）进行测试，有一个输入和一个输出或多个输出；每一个节点测试一个特征并将通过测试的标记传给它的全继节点；双输入节点对两个不同路径来的标记进行比较，如果它们满足产生式左手部中间元素的一致性，则将两个标记组成一个更大的标记；终止节点只接受那些使左手部成为示例的标记并输出模式匹配器的信息，从而改变冲突集。下面，用图2所示的两个元素同下列工作存储器中的两个元素进行匹配来说明 Rete 网的工作过程。

(Goal ↑Type Simplify ↑Object Expr17)

(Expression ↑Name Expr17 ↑Arg10 ↑OP * ↑Arg_2 X)

首先建立标记：(+ (Goal ↑Type Simplify ↑Object Expr17))，并将其发送到根节点处，根节点再将其传送给它的后继节点。图2中后继节点之一（右边）测试该标记后拒绝接受标记，因为标记的类名不为 Expression。而左边的节点类名正好为 Goal，与标记符合，因此它接受了这个标记，并将该标记传送给其后继节点，该后继节点测试 Type 值为 Simplify，同样也接受标记，并继续将标记传给它的后继节点。由于这时标记已到达双输入节点，而该双输入节点的另一个输入标记尚未到达，于是测试暂停，同时存储该标记，等待另一标记的到达。

当执行标记为（+ (Expression ↑Name Expr17 ↑Arg10 ↑OP * ↑Arg_2 X)）时，标记通过图2右顶节点到达双输入节点，这个双输入节点将新标记与前一标记进行比较，发现它们对同一变量的约束是一致的，因此该双输入节点建立并输出标记：

(+ (Goal ↑Type Simplify ↑Object Expr17)

(Expression ↑Name Expr17 ↑Arg10 ↑OP * ↑Arg_2 X))

当它的后继节点—Time 的终止节点接收到这个标记时，它就将 Time 这个示例 加到冲突集中。

通过对 PS 及 Rete 算法这两种计算模型较详细的分析，不难看出，它们在如下三个方面具有相同点：

1）模式都为可编译的；

2）目标都为常量；

3）目标集的变化相对地都较慢。

正因如此，Rete 匹配算法在 PS 解析器中得到了普遍的应用。

三、RIKBM 的设计方案

RIKBM（Rete Implemented in Knowledge－Based Machine）意为 Rete 算法在知识库机

中的实现。一般地说，PS 的并行实现要从并行算法和体系结构两方面来考虑，前者要求把 PS 解析器（Rete 算法）等映射到 PS 并行执行机器中去，后者则要求设计专用的 PS 并行执行机器。就并行算法来说，其主要差别在于并行粒度的大小，即在 PS 的哪一级上实现并行。PS 蕴含的并行性有：1）产生式级并行性（产生式间并行匹配和并行点火）；2）元素级并行性（条件元素间并行匹配和动作元素间并行处理）；及3）元素内部并行性（元素内部的项之间并行匹配）；就体系结构来说，它既取决于具体的并行算法，又取决于系统规模、处理器功能、作业通讯与调度、负载平衡以及互连拓扑结构等方面的因素。

设计 RIKBM 的目的在于硬化 Rete 匹配算法，高效地支持小规模 PS 的运行，并为进一步研制更实用的具有大容量的知识库机系统奠定理论和实践基础。

1. 设计依据

1）为方便用户，前端机将在 OPS_5 语言基础上用知识库操作语言 KBL 扩展知识库的建立、管理、维护和使用等功能；

2）以 Rete 匹配算法为基础，用后端智能存储器实现高效流水匹配操作，充分发掘 Rete 算法的并行性；

3）由于 Rete 网适合于数据驱动方式，RIKBM 将按数据流方式设计，即系统的运行将在各类包（数据元素包、产生式包、示例包和命令包等）的驱动下进行；

4）前后端机采用异步方式与同步高速来实现信息传送，系统负载平衡，通讯开销较小。

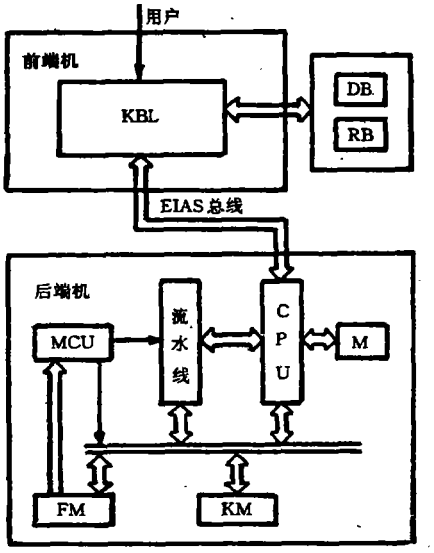

图3 RIKBM 总体结构

2. RIKBM 的总体结构

RIKBM 的总体结构如图3所示，其中前后端机采用 EIAS 总线连接；KBL 为知识库操作语言，它具有建库、查询、推理和维护等功能，并有良好的用户接口；RB 为产生式规则库，DB 为关系数据库，它们可由用户建立在磁盘中；后端 CPU 为32位处理器，其局部存储器 M 存放用于后端操作的一些管理程序等；流水线主要完成匹配工作；MCU 为存储控制器，实现对流水线及 KM 的控制，它根据 FM 的读出结果来决定流水线的操作方式，并决定从 KM 中读出的内容是送到流水线处理还是作为地址发送到 FM 和 KM 地址线上或直接送往后端 CPU 中；KM 中全部存放由前端机发来的已编译好 Rete 网；标记存储器 FM 存放由 KBL 提供的各类操作标记，其作用相当于一个索引表，旨在加速后端机的流水匹配操作。

3. 系统工作过程

RIKBM 的工作过程可分为初始化和运行两部分，如图4所示，用户首先在 KBL 下建立 PS 并装入 RB，然后 KBL 将其解释为 Rete 并以产生式包的格式用 DMA 方式批量发往后端机知识存储器 KM 中，而后端 FM 中则建立相应的操作标记。待初始化结束后，前端机便向后端机发送待匹配的工作存储器（WM）元素，当后端 CPU 接收到前端机发来的启动推理信号后，随即启动流水线进行流水匹配工作，并负责将匹配成功的示例集以示例包的形式发往前端机以便冲突归结。当前端机接收到由后端机发来的示例包后，便根据冲突归结（CR）策略从冲突集（CS）中挑选一个被匹配成功的产生式并执行其右手部动作，所产生的新的 WM 元素再由前端机以往后端机进行匹配等工作过程。

从宏观上看，在运行阶段，系统前后端机呈现在一条异步多功能流水线上。

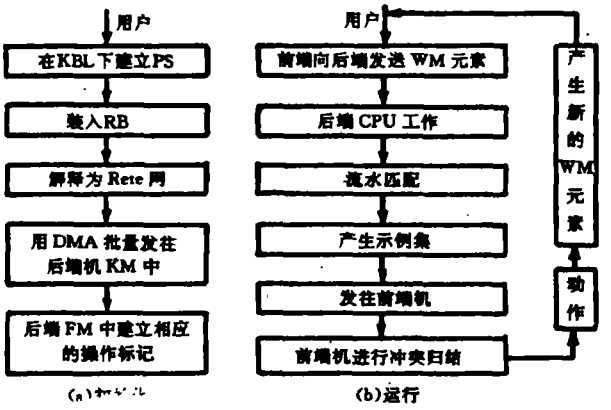

图4 RIKBM 工作过程

4. 前端机工作方案

前端机的逻辑结构如图5所示，其核心为KBL，它具有建库、查询、推理和维护等功能，具体可用下面的 KBL 文本框架来表示。

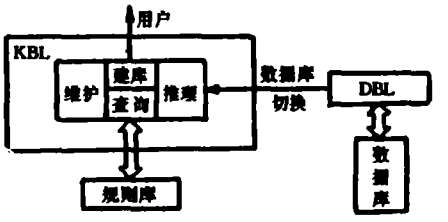

图5 前端机逻辑结构

KBL 框架如下：

设 RB 为产生式规则库，DB 为数据库，parameters 为进行问题求解的参数。

(1) Structure RB－name ；建立一个 RB 结构
(2) Add－to RB－name ；向一个 RB 增加一条知识
(3) Delete－from RB－name ；从一个 RB 中删除一条知识
(4) Modify RB－name ；修改 RB 中的知识
(5) RB－name [conditions] ；查询某些条件下 RB 静态内容
(6) Switch－to DB ；切换到 DB 系统状态
(7) Open－RB ；打开 RB，准备使用
(8) Make [parameters] ；准备推理条件
(9) Conflict－resolution ；设置冲突归结策略
(10) Run [parameters] ；启动推理

另外，前端机为用户还提供了一个透明的 OPS5接口环境，用户可按照原 OPS5解析器的约定进行编程和调试等，但增加了包编译的额外工作。用户编好的 OPS5程序一旦通过语法检查，即可进行包编译工作，其结果是将程序中的各类参数生成相应的参数表，并以产生式为单位划分成包，然后前端机将一个 或几个包传送到后端机 KM 中，这样便可将后端机的匹配模式确定下来，系统的运行过程是通过前端机向后端机发送 WM 元素来将 PS 中的“匹配→选择→动作→匹配→……”这一循环操作运行起来。

5. 后端机工作方案

后端机包含接口与匹配部件两个部分。接口部件的设计依据是以异步方式及同步高速来

实现前后端机的信息传送。所谓异步方式是使得前后端机之间随时可以由一方主动发信息，而由另一方来接收，这个过程相当于握手通讯协议，其优点在于能确保所发信息能够成功地转交给对方而不丢失。所谓同步高速是采用 DMA 批量传送方式并以系统访存的速度来进行信息传送。

图6为接口部件示意图，其具体信息交换将遵循如下几条协议：

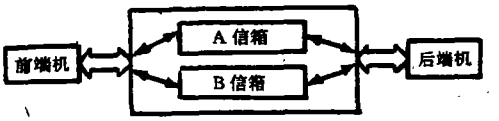

图6 接口部件示意图

1）前后端机各自占有一个信箱，两个信箱 A 和 B 的占有是互斥的，双机以常规的访存动作来读写（取函投函）自己所属的信箱；

2）双机之间逻辑上不存在不请求信息而只有发送信息的情况。请求信息的过程只是在发送方发送信息之后，由接收方解释接收到的命令并向发送方发送所需信息的过程；

3）当有一方欲将信函交给对方时，则首先将信函投入信箱，然后提出交换信箱的请求，并由对方从所属信箱取出信函；

4）为避免对方正在使用信箱而不希望交换信箱以致于造成有用的信息可能丢失的情况发生，在信箱交换过程中，采用异步握手信号，即前后端机都不允许在投函取函时进行信箱交换。

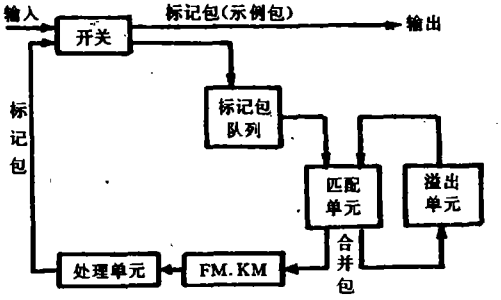

图7 匹配部件功能图

匹配部件的设计是以加速匹配过程为准则，图7所示的匹配部件功能图是一种数据驱动的单环结构，具有很强的流水线能力和高度的并行度。环中的诸部件组成流水线的各个级，信息以标记包的形式沿环定向传递。标记包可经由开关插入环中或将计算结果由环中取出；标记队列中保存了前端机预处理后所设的大量标记；匹配单元合并在沿途中具有相同标记的标记包；失配的标记由溢出单元处理；FM 和 KM 存储器将构造完全可以执行的操作包；处理单元执行所要求的操作，最终又产生出一些新的标记包，然后又开始另一新的匹配循环，一旦有示例包产生，立即由后端 CPU 通过 DMA 通道发往前端机。在该数据流环中，各部件仅通过向环发送的标记包相互作用，因而它们可以独立地完成各自的功能，该功能可以相互重迭甚至允许彼此赶超。

四、结 束 语

通过对知识库机的研究及一个原型系统 RIKBM 的设计，笔者认为，PS 作为演绎知识的核心，其表示方式很接近人类的思维方式，因而有着广泛的应用前景，但由于实现其解析器的 Rete 匹配算法受限于现有的硬件环境，故在知识库机上实现 Rete 算法是提高 PS 运行效率的有效解决方法。再者，专用体系结构的知识库机在目前仍不失为知识工程研究和发展中一个十分需要的部分，它可望能解决知识推理、管理和使用的关键效率问题。

（下转第35页）

善了OOFOX系统。所开发的概念财务软件系统充分细划了财会领域中的各概念，刻划了各概念之间的内在联系，使系统的软件构件重用与系统的维护均按照领域概念进行，操作简单，轮廓清晰。OOFOX系统对广大熟悉DBASE、MFOXPLUS系列数据库管理系统的微型计算机用户来说，不失为一有效的实用系统。

参 考 文 献

〔1〕 S. B. Zdonik and D. Maier (eds.), Readings in object—orinented database systens, CA: Morgan Daufmann Publishers. 1990
〔2〕 张国锋，面向对象的程序设计和C++语言，北京科海培训中心，1991. 1
〔3〕 周苏等，FOXBASE+及其程序设计技巧，天津科学技术出版社，1988. 11
〔4〕 徐金梧等，Turbo C使用大全 V1.5—V2.0，北京科海培训中心，1990. 4

（上接第15页）

参 考 文 献

[1] C. L. Forgy, OPS5 User's Manual, TR·CMU—CS—81—135, Carnegie—Mellon Uni., 1981.
[2] C. L. Forgy, Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, AI 19, pp. 17—37, 1982.
[3] 胡铭曾、张学海、王文敏，产生式系统的并行模型，全国人工智能会议论文集，1989.
[4] T. Ishida and S. J. Stolfo, Towards the Parallel Execution of Rules in Production System Programs, Int. Conf. on Parallel Processing, pp. 568—575, IEEE, 1985.
[5] D. I. Moldovan and M. F. M. Tenorio, Mapping Production System into Multiprocessors, Int. Conf. on Parallel Processing, pp. 56—62. IEEE, 1985.
[6] 张学海，产生式系统并行点火的研究，硕士论文，哈尔滨工业大学，1988年12月.
[7] 张学海，产生式系统并行处理现状，电脑学习，1989年第6期.
[8] 方滨兴，包驱动执行机制的研究与实践，博士论文，哈尔滨工业大学，1989年6月.