

支持产生式系统并行推理的动态推理网

应晶 何志均

(浙江大学人工智能研究所 杭州 310027)

摘 要

产生式系统作为人工智能领域的重要分支,已得到较为成熟的应用,同时也不断面临一些新的需求.针对产生式系统并行处理能力的需求,提出一种基于动态推理网(DRN)的知识库构造方法,并介绍整个知识库的描述与构造方式,以及相关算法的实现.在此基础上支持产生式系统的并行推理功能.

关键词: 产生式系统,并行推理,规则库.

1 产生式系统的推理机制

随着产生式系统应用的深入,产生式系统面临着两个问题:一是在系统建立的初期,由于知识库比较小,加上设计人员对初始知识模型的推敲和精心构造过程,整个知识库能够保持一致,但随着时间的推移和新规则的不断加入,知识库变得复杂庞大,此时知识库中各规则单元之间的相互影响与联系随之变得复杂和难以捉摸,因此容易导致不一致性^[1];二是随着知识库的增大,用户对系统的推理效率也有了更高的要求,包括启发式搜索、回溯等方面^[2],进一步需考虑并行推理的功能^[3],而且在知识库的组织上提出新的需求,即用户对整个系统的推理过程、规则的激活及推理路径均需有一个全局的把握.为了解决这些问题,本文在产生式系统中借鉴 Petri 网的特性提出一个动态推理网模型.利用此模型可以对整个知识库中的产生式规则进行有效的组织,不仅能直观地反映系统的整个推理过程,而且支持并行推理功能,在知识库的一致性检查方面也提供了更好的支持.

2 动态推理网(DRN)

产生式系统有以下两个特点:一是在产生式系统中由于规则及其激活过程的动态性,使得整个推理过程具有动态性和不确定性,需要一种机制来描述其中的动静态特性;二是规则库中的每条规则可形式化地描述为

$P(L, R, K, C)$ 其中 L 是前提, R 是结论, K 是规则的激活界面, C 是推理过程包含的上下文.这种描述结构与 G. A. Petri 所提出 Petri 网有一定的类似性. Petri 网对于研究系统构成模型、系统的数学描述及了解系统动静态特性来说是一种有效的工具,文[4]针对产生式系统研制了一个基于 Petri 网的知识库一致性检查系统.因此借鉴 Petri 网的描述机制,作者提出了一种动态推理网结构.其主要思想是对知识库构造一种静态结构,

并借助于推理上下文和激活界面,较为清晰地描述整个动态推理过程。下面介绍动态推理网 (DRN, Dynamic Reasoning Net):

$$DRN = (P, EP, ET, GC)$$

其中 P 类似于 Petri 网中的 place, 代表规则所感知的谓词集 p_i , EP 是 P 的扩充。对一个包含 K 条规则的产生式系统而言,有 $(ep_{i1}, ep_{i2}) \in EP$ 且 $(ep_{i1}, ep_{i2}) \rightarrow rule_i$, 即 $ep_{i1} \rightarrow L_i, ep_{i2} \rightarrow R_i, (i = 1, \dots, k)$ ET 是一种触发器(即推理上下文)。类似于 transition, 并在原有基础上增加: (i) 当前触发过程的触发时序数(即 ep_{i1} 所感知的 place 数), 用于冲突解决; (ii) 当前触发所对应规则激发的初始优先权。 GC 是门电路结构。对每一个 ep , 存在一个细粒度的扩展结点。此结点用门电路的方式加以组织, 形成直观的由谓词值触发的时序电路。 GC 中包括三种描述机制: 与门、或门、非门。考虑下面一个产生式系统的规则库:

- rule 1. if p_1 and p_2 then p_3
- rule 2. if p_4 and (p_5 or p_6) then p_7
- rule 3. if p_1 then p_5 and p_7
- rule 4. if p_7 and (not p_8) then p_9
- rule 5. if p_9 and p_4 then p_{10} and p_3

根据上述定义, ET 的触发条件和触发结果利用门电路 GC 的描述见图 1。

采取 DRN 来描述整个知识库,有以下优点:

- a) 利用门电路中的丰富描述机制及较强的时序性(触发机制)充分地组织规则;
- b) 采用粗/细粒度 (EP 和 GC) 的结构来刻画规则的特性;
- c) 利用每个 EP 的 MISO (多人单出) 机制设置一致性检测器检查输出位值的不一致性;
- d) 同一推理周期允许多条规则激活执行,直接支持了并行推理机制;
- e) 用户对整个推理过程有全局的把握,为各类解释、调试工作带来便利。

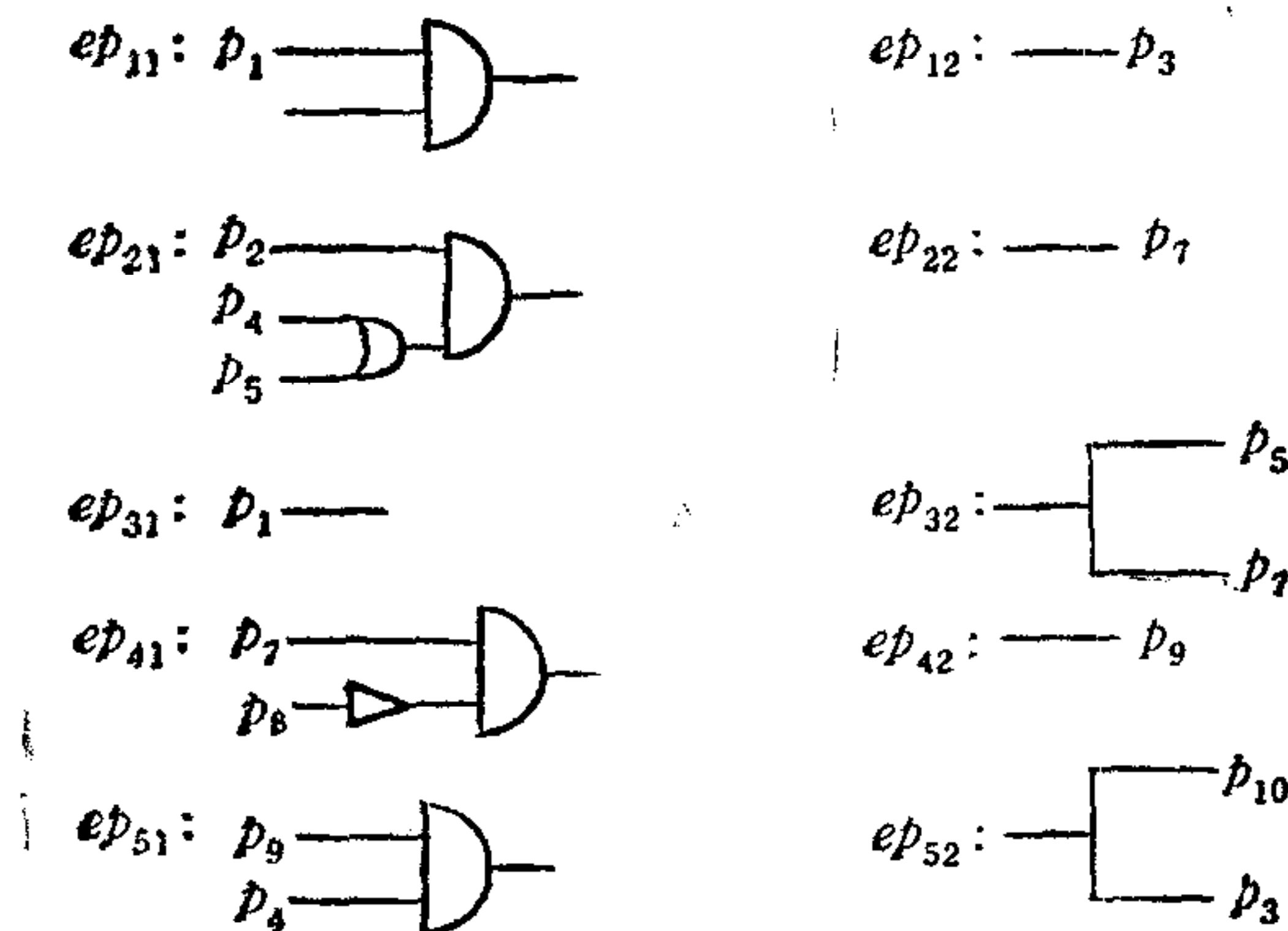


图 1 规则的门电路描述

3 数据结构与算法实现

DRN 中最主要的数据结构描述包括:

```

/*EP 结构 */
struct Extended_Place {
    int      index;          /*索引值*/
    int      type;          /*规则的前提或结论*/
    int      truth_value;   /*真值*/
    int      chron_num;     /*触发时序数*/
    int      cycle;        /*当前推理周期*/
    int      place_link[MaxPLACE]; /*感知的所有 place*/
    struct Extended_Place * ep_pair; /*指向与此 ep 组成同一规则的另一 ep*/
    struct Gate_Circuit * gc_graph; /*细粒度的门电路结构*/
}

/* E T结构*/
struct Extended_Transition {
    int      index;          /*索引值(即规则名)*/
    int      place_link1 [MaxPLACE]; /*EP 前提感知的 place*/
    int      place_value1 [MaxPLACE]; /*place 的真值*/
    int      place_link2 [MaxPLACE]; /*EP 结论感知的 place*/
    int      place_value2[MaxPLACE]; /*place 的真值*/
    int      cycle;        /*当前推理周期*/
    int      chron_num;     /*触发时序数*/
    int      (*eval_fun)(); /*根据时序数及优先权的评价函数*/
    int      init_priority; /*初始优先权*/
}

/*门电路结构 GC*/
struct Gate_Circuit {
    int      index;          /*索引值*/
    int      place_node;     /*单结点(真值)*/
    int      truth_value;   /*当前 GC 图的检测值*/
    struct Gate_Circuit * And_Part; /*"与门"部分*/
    struct Gate_Circuit * And_Next; /*"与门"的其它连接*/
    struct Gate_Circuit * Or_Part; /*"或门"部分*/
    struct Gate_Circuit * Or_Next; /*"或门"的其它连接*/
    struct Gate_Circuit * Not_Part; /*"非门"结点*/
}

```

在系统中牵涉三个主要的算法: 并行推理算法、门电路真值检测算法及一致性检测算法。

门电路的真值检测采取如下原则: 1)对与门进行负值检测后,取反;2)对或门进行正值检测;3)对非门进行正值检测(算法描述略)。

一致性检测算法:

```

BOOLEAN Consistency_Check /* 初始化 token = TRUE*/
{
for 每个  $p_i$  (临时工作区)  $i = 1, \dots, k$  do {
检测其真值的传递来源:  $ep_{i2}$  的 place_link 值
  若 place 数目为 1, 则成功, 跳过;
  否则, 对  $p_i$  的各来源值检测, 如一致则  $p_i \Rightarrow$  临时工作区
      否则 token 置 FALSE, 出错
}
若 token = TRUE, 则置更新开关=1, 临时工作区  $\Rightarrow$  产生式系统工作区
否则置更新开关=0, 报告不一致性情况.
}

```

并行推理算法:

步骤 1. 创建规则库中所涉及的工作区基本单元, place, $\{p_i\}$;

步骤 2. 对每一条规则 $rule_i \in R$, 建立扩展 place, ep_i ;

$(ep_{i1}, ep_{i2}) \rightarrow rule_i, ep_{i1} \rightarrow L_i, ep_{i2} \rightarrow R_i$;

步骤 3. 构造用于真值检测的全局 transition, TD_{ei} ;

步骤 4. 对每一 ep_i , 构造细粒度的门电路结点, 用 Gate_Circuit 数据结构表示;

步骤 5. 构造扩展的 transition, 设置初始优先权及本身的触发时序数

步骤 6. 建立产生式系统工作区与整个 DRN 的接口 (用工作区中的数据值传递给 p_i , 并触发 ep_{i1});

步骤 7. 建立并初始化针对临时工作区的一致性检测器 CD_{ei} (即全局的 transition) 及相应的更新开关 (在一个推理周期后, 并行执行的几条规则对工作区的修改可能出现不一致性, 这样 CD_{ei} 禁止当前某一 p_i 的不同值通过, 并关闭更新开关);

步骤 8. 当运行环境就绪后, 由初始工作区激活整个推理过程. 对每一推理周期重复: 1) 采用 Gc_Truth_Check 算法, 并行检查每个 ep_{i1} 的真值; 2) 采取基于时序触发的冲突解决, 根据 ep 的门电路中入口的触发时序数 (感知的 place 数) 决定其匹配程度, 再进行 $\min(\text{init_priority}, \text{chron_num} * \text{因子}, \text{eval_fun}())$ 运算. 在整个扩展动态网中, 由于支持并行推理机制, 因此在计算触发优先权时采用多路激发的冲突解法. 3) 并行执行被激活的多条规则; 4) 采用 Consistency_Detect 算法 (出现不一致性时, 对相关规则的 et_i 由用户交互设置其优先权为 -1), 完成工作区的更新操作; 5) 若扩展的 DRN 保持稳定 (没有新的规则触发或工作区满足目标状态), 则整个推理过程结束; 否则转 1).

4 讨论

利用动态推理网 DRN 构造产生式系统中的规则库及其推理过程, 充分体现了基于知识的推理的动态性和不确定性, 显示出多个视图和自动的激发状态. 用户只需控制初始的构造和点火效应, 则产生式系统规则库所激发的一系列结果均能检测, 机理简单. 利

用门电路的特点提供了粗细粒度两个层次的结构,较好地模拟了自动的触发时序。

在此基础上系统反映的一个重要的特性是支持的并行推理机制。工作区与推理网的动态匹配为同一周期中的多条规则激发提供了基础。采用多路激发的冲突解决及在一个触发过程上所附的优先权计算能较为容易地实现这一点。此外,系统在一致性检查上利用网络传输端的检测器提供了更为完善的机制,而且避免了产生式系统中本质的回溯问题。同时用户对推理过程及工作区状态有清晰的把握,增强了系统的解释机制及调试功能。

作者对并行推理进行了模拟执行。在同一推理周期中,多条规则的执行实际上有着时间先后关系(按优先权排列),这样在一致性检测器上记录着当前推理周期所产生的每个 p_i 的位值情况。从运行情况看已较好地接近了并行推理的功能。DRN 对规则间的联系不是很深入,因此有必要对每条规则的门电路之间进行调整,建立恰当的连接,以便把某些问题归结为支持电路检测和门电路最优化设计问题^[5],甚至可以考虑硬件实现的产生式系统推理结构。这些工作有待于进一步的研究。

参 考 文 献

- [1] Nguyen T A, et al. Checking an expert systems knowledge base for consistency and completeness, Proc. IJCAI' 85, 375—379.
- [2] Dixit V V. Minimal state space search in parallel production system, Proc. Int'l Conf. on Parallel Processing, 1989, 255—259.
- [3] Oflazer K. Partitioning in parallel processing of production systems, Proc. Int'l Conf. on Parallel Processing, 1984, 110—115.
- [4] 应晶,吴朝晖,何志均. 基于 Petri 网的知识库一致性检查系统,计算机研究与发展, 1992, 29(8): 30—35.
- [5] Gupta G, Jayaraman B. A model for combined and-or parallel execution of logic program, Proc. Int'l Conf. on Parallel Processing, 1989, 260—263.

DYNAMIC REASONING NET TO SUPPORT PARALLEL REASONING IN PRODUCTION SYSTEM

YING JING HE ZHIJUN

(Artificial Intelligence Institute, Zhejiang University Hangzhou 310027)

ABSTRACT

As an important aspect in the domain of artificial intelligence, the production systems of AI is confronting some new requirements and challenge. Aiming at the demand of parallel processing for production systems, this paper puts forward a kind of dynamic reasoning net (DRN) based generation method for knowledge base, and describes the entire constructive procedure and the implementations of three relevant algorithms. On this basis, the parallel reasoning function of production systems can be supported by this method to some extent.

Key words: Production system, parallel reasoning, rule base.