专家系统 DIES 知识库 (16)

计算机科学1995Vol. 22№. 2

76-78,81

DIES：一个领域独立的专家系统

褚东升 TP18

（山东财政学院信息管理系 济南250014）

摘 要 In this paper, the structure and the functional features of the DIES system are described. DIES represents an intelligent problem solver based on both declarative and normative knowledge. It is able to combine backward and forward inferences and to explain the derived reasoning line as well as the causes of features. Also, it can modify and yet keep consistent the knowledge it accumulates.

关键词 Expert system, Problem solver, Domain independent.

一、引 言

DIES 系统是一建造基于知识的自动问题求解器的领域独立专家系统，在该系统中其描述性知识和规范知识（规则）均为知识库中的标准类型项。DIES 系统具有双重结构，即认知系统和求解系统，并可满足下列要求：

（1）知识库是按照各自的领域及系统特定的准则组织的；

（2）能够进行知识积累（扩充、修改、知识库一致性的自动维护）；

（3）支持建立在继承和一般类型推理基础上的面向检索的推理；

（4）具有在类 PROLOG 推理机制基础上的演绎推理能力，具有回溯功能；

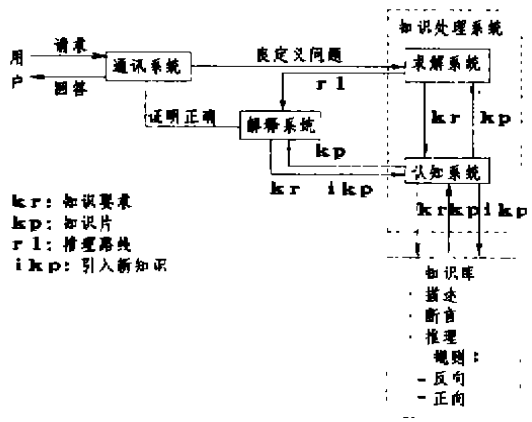


图1 DIES 系统的结构框架

（5）可以进行由正向链支持的间接推理；

（6）在问题分解期间能够进行规划求精；

（7）可根据用户要求，对推理路线进行解释或改变。

图1给出了 DIES 系统的结构框架示意图。

DIES 系统是用 LISP 语言编写实现的，结合适当的知识，该系统已作为工具初步建立了设计领域中的专家系统。

二、知识库

DIES 系统的知识库中用于表示实体的基本结构单位称为描述项，对应于实体的标准类型。如对象、过程及规则；描述项又被分配到几个不同字典中。

实体类中的对象和过程共用以下的语法结构：

```
(entity-name
(GENERAL CHARACTERISTICS
  (CATEGORY      —对象//过程)
  (CLASS    —分类链)
  (SORT     —标识符，作用在知识片上容许的操
             作符集)
  (PROTOTYPE      —是/否)
  (COMPLEXITY     —原子的/复合的)
  (EXTERNAL STRUCTURE
       —实体相联系的关系)
       —避免产生不合理组合的限制接受模式)
  (INTERNAL STRUCTURE      —成分间的关
                                系)
  (PROPERTIES     —其它特征))
```

表示实例值（PROTOTYPE NO）的实体从属于几个不同的描述项，称为公开项并由不同的 CLASS/SORT 偶对表示。与知识库有关的认知功能集包括：

（1）知识获取功能；

（2）与现存实体（知识片）有关的功能：在描述

项中增加特性；·删除一实体；·删除描述项中的特性；·修改描述项。

(3)与知识库一致性有关的功能：·基于分类链及其它关系链的隐含一致性；·基于对每一过程类型的自动过程调用的由用户提供的一致性。

就过程而言，其描述项的内容应适于直接模式调用的需要。过程中的 CATEGORY 是由能自动调用它的认知或求解处理类型给出的。根据调用者的不同性质，一个过程可能具有多个过程体。EXTERNAL STRUCTURE 为相应的调用模式提供了保护，过程体本身包含在 INTERNAL STRUCTURE 中，而在 PROPERTIES 描述符下则给出了过程所在的环境，即前置、后置及连续条件。这些环境用来解决由直接模式调用机制所引起的冲突现象。

知识库中包含两类推理规则，即反向规则和正向规则，分别支持目标驱动下及事件情况驱动下的推理。反向规则的构成形式为：

(rule-name((conclusion)
(premise-1)…(premise-n))

反向规则的语义解释是基于逻辑程序设计中的两个知识积累解释的，即：

(1)说明性解释：结论的似然性直接与前提的似然性相关；

(2)面向问题归约的解释：结论表示一个要求解的问题或一个要完成的方案，而前提则表示相应的子问题、子目标或子方案。

正向规则的形式与反向规则的形式类似，其形式为：

(rule-name ((situation)
(action-1)…(action-n))

正向规则的解释象通常所施加于"情况-动作"规则的解释一样，每当在当前知识状态下遇到特定的"情况"时，系统将依次执行其后的"动作"操作。这些"动作"操作包括：

(1)在当前知识片集合上进行隐含的修改操作；

(2)系统驱动的对话，从而使得从用户方面获得更多的支持；

(3)在给定的环境下处理有关新目标的决策或推荐出其它的动作。

当达到一个新选定的目标后，全部的解将根据行动的集合依次被求解，"结论"及"情形"则分别由与问题描述语句或当前知识状态相匹配的模式来表示。

在 DIES 系统中，反向推理和正向推理是交叉进行的，后者对前者完全起着一个辅助作用，达到一个给定目标的推理，主要依赖于反向推理过程，这相当于一递归目标归约程序，而情形驱动的推理则只是起到预见的作用，然而却是一个很有效的信息供应者，从而扩展了当前的知识范围。由于在目标驱动反向推理基础上产生的子目标实现起来更为直接，因此达到给定目标的推理也变得更加容易。

三、基本求解器

通常，一个求值(搜索)模式的构成主要由下列因素决定：

(1)问题空间的求值方向(反向、正向、双向)；

(2)用以确定在每个阶段下一个可用方案的优先选择次序的冲突解决策略；

(3)回溯策略，用来决定回退到上一步以便重新开始沿一新方案求值。

DIES 的基本求解器实际上是一个可提供自动回退的 PROLOG 的反向归约机制，求解过程是由下列形式的语句开始的：

(RESOLVE((goal-1)…(goal-n)))

这里(goal-i)可由任意模式表示，整个右端的目标合取式对应于要求解的问题。求解目标的处理是以类似于堆栈的方式从左至右一次处理一个目标进行的，求解处理包括两个阶段：

(1)搜索：系统在知识库中可通过直接或基于继承的检查来达到目标要求；

(2)推理解释：在搜索失败的情况下，系统"重写"目标，即根据反向推理规则把目标分成几个子目标。

推理解释阶段是对反向规则集合应用"认别-重写"循环不断地进行改写，直至没有规则可用或子目标已满足要求为止，其求解处理的算法为：

```
* * solve(GOAL,rules,varbind)
//GOAL:被求解目标的合取//
//rules:从知识库中选取的规则子集//
//varbind:记住变量的束定供以后在回答和解释中使用//
        FALL←false;
while   GOAL≠NIL & not FALL do
if      the first subgoal is directly retrieved
then    * update(varbind);
        GOAL← * var-update(rest(GOAL));
        * * solve(GOAL,rules,varbind);
or      the first subgoal is a solving procedure
then    * execute the procedure;
        * update(var-bind);
        GOAL← * var-update(rest(GOAL));
        * * solve(GOAL,rules,varbind);
else    //开始推理解释//
        i←1 & SOLVED←false;
while   i<the number of rules & not SOLVED do
    var1← the first subgoal &
```

```
    var2← conclusion of rule(i);
if      * unify(var1,var2);
then    GOAL←APPEND( * var-update(premis-
    es of rule (i)))
    ( * var-update(rest(GOAL)));
     * update(varbind);
    if * * solve(GOAL,rules,varbind);
    then SOLVED←true;
    else i←i+1;
else    i←i+1;
;
;
if    not SOLVED
then    FALL←true
;
answer(FALL);
```

四、正向推理

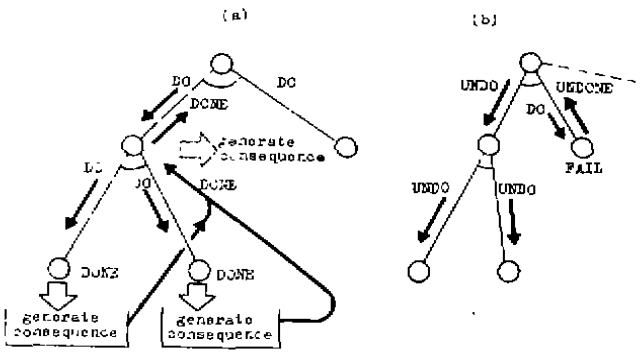
根据用户的显示选择，求解可以双向进行，因此每当达到一个子目标时，为了及时得到中间结果，可以插入正向推理，系统将根据以下的因素来执行这类推理：

(1)当前的知识状态；

(2)当前处理过程的进展情况；

(3)下一步要进行的新处理。

系统的最终目标是支持反向推理，增加知识积累，这样可使还没有达到的子目标更容易达到。正向推理是建立在"识别执行循环"基础上的，即在正向规则集中使用这种识别执行循环直到不再有任何可进一步使用的规则为止。循环开始时所使用的初始"情形"是由最新达到的目标表示的，其结果可能根据出现的新"情形"而再次激活其循环。作为正向推理递归展开的结果，知识库将受到一些隐含的修改，在特定环境下(如对话情况)某些事实还需要对其进行精确的定义或证实。证实可能是完全隐含的，这将完全依赖于系统，由系统进行求解处理，而其目标即为给定事实，证实也可以是半隐含的。证实不仅依赖于系统，其证实信息也可以由用户提供。当某种"情形"与多个正向规则相匹配时，所有这些规则均被执行之，这是因为以这种形式蕴含的所有结果都可能会对以后的推理处理有用，这一过程可由图2所示。



(a)运用反向归约及正向推理成功所生成的解释树

(b)在前级推理失败情况下的语义解释树

图2

五、解释器

DIES 中的解释机制为用户提供了沿推理轨迹回溯的可能，从而可对系统解答的正确性进行检查。解释器的另一个功能则是能够给出求解处理过程中失败的原因，另外，一些不可解的目标或子目标以及一些可能解的建议也将给予记载，使得求解操作变得更加可靠，也使得用户在求解处理过程中能够更加起到协同的作用。

解释系统还可以用来完成下述功能：

(1)教育：这是由于用户可能会提出一些问题并将其答案与系统给出的答案进行比较；

(2)知识库的调整：因为在推理处理的进程中知识库中的一些错误知识可能会反映出来，这就需要对知识库进行动态地修正。

为了满足用户的某些需求，系统中允许 HOW、WHY 和 WHEN 命令根据以下一些基本要求给出某些剪裁解释：

(1)系统可以允许用户自己选择其要解释的推理树的深度，所提供的解释的详细程度可足以满足用户的要求；

(2)完备性：系统应能够验证其推理的每一步行为都是合理的，除非已进行的处理有负作用。在这种情况下，解释系统要保存其在知识库上进行的所有修改轨迹。

六、进一步的工作

在现阶段，我们已开发完成了 DIES 系统的第一版本，进一步的工作将在第一版本的基础上增加一些新的功能，主要有以下几个方面：

(1)特殊规划功能：用来设计改进冲突的选择策略；

(2)回溯检查功能：用来支持回溯分析的智能开发；

(3)获取与学习功能：用来对求解处理进行有效的支持。

(下转第81页)

$$S_{ij}=\begin{cases}\alpha_i(-b_i+a_{ii}) & i=j\\ (\alpha_i a_{ij}+\alpha_j a_{ji})/2 & i\neq j\end{cases}$$

那么式(3)的平衡点是指数稳定的。

证明：(提示)取 $v(x)=\sum_{i=1}^{N}\frac{1}{2}\alpha_i x_i^2$ (11)

为式(3)的 Laypunov 函数，显然 $v(x)$是正定的，因 $\alpha_i>0$。沿(3)式的解求 v 的导数，再利用定理2，即可得证。 □

注意：i)在文[1]中，要求 $G_i(x_i)$连续可微，事实上，只需 $G_i(x_i)$分段可微即可，可以降低约束条件。利用类似的方法，在 $G_i(x_i)$为连续函数的条件下得到平衡点是稳定的所应满足的条件。ii)对于定理5的条件 b)，只需验证对称矩阵 S 的最大特征值是负的即可，这可以利用文[4]求对称矩阵的最大特征值的估算式。

3. 不稳定结果

定理6 设 $N_1\neq\varnothing$(非空)，$N_1\subset N_0$，$N_0\triangleq\{1,2,\cdots,N\}$，$B_i(r_i)$为 $x_i=0$的某邻域，如果以下条件成立，

a)存在常数 $a_{ij}\in R(-\infty,+\infty)$，使对所有 $x_i\in B_i(r_i)$，$x_j\in B_j(r_j)$，$i,j\in N$，均有，

$$-x_i\sum_{j=1}^{N}A_{ij}G_j(x_j)\leqslant|x_i|\sum_{j=1}^{N}a_{ij}|x_j| \quad i\in N_1$$

$$x_i\sum_{j=1}^{N}A_{ij}G_j(x_j)\leqslant|x_i|\sum_{j=1}^{N}a_{ij}|x_j| \quad i\in N_0-N_1$$

b)存在 n 维向量 $\alpha=(\alpha_1,\alpha_2,\cdots,\alpha_N)^T>0$，使检验矩阵 $S=[S_{ij}]$为负定的，其中：

$$S_{ij}=\begin{cases}\alpha_i(b_i+a_{ii}) & i=j\in N_1\\ \alpha_i(-b_i+a_{ii}) & i=j\in N_0-N_1\\ (\alpha_i a_{ij}+\alpha_j a_{ji})/2 & i\neq j\end{cases}$$

那么：

(i)(3)式的平衡点($x=0$)不稳定，当 $N_1\neq N_2$ 时；

(ii)(3)式的平衡点($x=0$)完全不稳定，当 $N_1=N_0$时。

证明：(提示)令 $v(x)=\sum_{i\in N_0-N_1}\frac{1}{2}\alpha_i x_i^2-\sum_{i\in N_1}\frac{1}{2}\alpha_i x_i^2$，沿(3)式的解，求 $v(x)$的导数，并利用以上条件 a)化简，然后利用定理3，即可得证。 □

参考文献

[1]A. N. Michel et al., Qualitative analysis of neural network, IEEE Trans. CAS, vol. 36, №. 2, p229—p243, 1989

[2]A. N. Michel 等著，郑应平译，大规模动态系统定性分析，辽宁科学出版社，1985

[3]但琦、童颖，非对称互联神经网络平衡点稳定性分析，计算机科学 Vol. 20, №. 1, 1993

[4]但琦等，Hermite 矩阵最大(α)特征值的估算，后勤工程学院，№. 1, 1992

(上接第78页)

同时，我们也正在进一步探索新的应用领域，如用于自动专家技术故障诊断等领域。

主要参考文献

[1]Cohen P. R. and Feigenbaum E. A (eds.), The Handbook of Artificial Intelligence, Vol. Ⅱ, William Kaufmann, 1982

[2]McDermott, J., R1: A Rule-Based Configurer of Computer Systems, Artificial Intelligence, №. 19, 1982

[3]Stefik, M. et al., The Organization of Expert Systems, A Tutorial Artificial Intelligence Vol. 18, №. 2, 1982

[4]Weiner J. L., BLAH: A System which Explains Its Reasoning, Artificial Intelligence, №. 15, 1980