

知识库推理和维护系统(KBIMS)

杨 莉 胡守仁

(电子计算机系)

摘 要 赋予知识库自动推理和维护功能,从而使知识库有效和实用,是知识库面临的关键问题。本文提出了一程基于缺言推理的知识库自动推理和维护方法,并介绍了根据此方法,在 VAX-I/780 上使用 GKD-Prolog 作为工具实现的一个知识库推理和维护系统。

关键词 人工智能,知识库,Prolog,非单调推理,缺言理论,知识库推理和维护

分类号 TP39

目前,几乎所有的人工智能系统都有一个庞大的知识库。如何保证知识库具有无冗余性和一致性,如何使知识库具有自动推理和维护能力已成为人工智能系统达到高效和实用所面临的一个关键问题。

传统的知识库系统是建立在谓词逻辑基础上的单调系统,即知识库中已知为真的知识数目随时间而严格增加。这是由于新的知识可由外界加入系统,也可由系统推出,但这种加入和推出决不会导致系统中已为真的知识变为无效,因为它总假定存贮在知识库中的知识是有效的。当一个知识要加入到知识库中时,仅进行如下检查:

- (1) 可证明性检查 (provability check);
- (2) 矛盾性检查 (Contradiction check);
- (3) 冗余性检查 (Redundancy check);
- (4) 独立性检查 (Independency check)。

仅当要加入的知识不能被当前知识库所证明(推出),并且加入知识库后不引起矛盾和冗余时才消化 (Assimilation) 进入知识库中。

然而,由于人对客观世界的认识不仅是不确定的,而且往往也是不完全的,因此,基于人类知识的智能化知识库系统所拥有的知识应是过渡性的,还需要不断地进行自我改进和完善。它应允许不断地用新的知识取代已存在的知识,对已存在于知识库中的知识进行修改,即应有如下功能:

- (1) 更新和修改知识库中已存在的知识;
- (2) 冗余的消除;

(3) 由于发现矛盾而恢复知识库。

根据上述观点, 本文给出一种基于缺言理论来实现知识库自动推理和维护的方法。

1 基本概念和定义

使用基于缺言理论的非单调推理技术来实现知识库的自动推理和维护, 其基本思想是, 在知识不完全或知识有限的情况下, 做出与系统当前知识相一致的假设, 并使该假设成为系统当前的信条, 而当系统中出现不一致或矛盾时, 采用相关性回溯策略对这些假设加以检查和修订, 以使系统恢复一致性。下面给出缺言理论的定义。

一个缺言理论定义为二元组 $\Delta = (D, Q)$, 其中 D 是应用于某个模型化世界的缺言规则集, Q 是该世界的知识集, 亦即一阶公式集, 缺言规则的一般形式表示为:

$$\frac{\alpha(X); M\beta_1(X), \dots, M\beta_n(X)}{W(X)} \quad (1)$$

其中 $\alpha(X), \beta_1(X), \dots, \beta_n(X)$ 和 $W(X)$ 都是一阶公式, 其自变量 $X = X_1, \dots, X_m$ 。(1) 式的含义是, 对所有个体 X_1, \dots, X_m , 如果 $\alpha(X)$ 被相信, 且 $\beta_1(X), \dots, \beta_n(X)$ 与知识系统 Q 相容, 则 $W(X)$ 可以被相信。这里我们称被 Q 所相信的知识为 Q 的信条, 例如:

$$\text{bird}(X); \text{fly}(X) / \text{fly}(X)$$

是一个规则, 则第二个 $\text{fly}(X)$ 是一个信条。在(1)式中 $\alpha(X)$ 称为前提, $\beta_1(X), \dots, \beta_n(X)$ 称为合理性条件, $W(X)$ 称为结论。这里合理性条件只是一种假设, 即 $\beta_i(X), i = 1, \dots, n$, 成立并不导致当前的 Q 出现矛盾。这里所说的“当前”是有意义的, 因为在用合理性条件进行推理时, 以后得出的结论可能会使 Q 出现矛盾。为了在系统中能用缺言理论来实现知识库的自动推理和自动维护, 我们给出几个基本定义和约定。

定义 1 事实和规则统称为信条 (即知识)。每一信条又对应一个结点, 用 n_i 表示, $i = 1, 2, \dots$, 用 N 表示系统中所有结点的集合。

定义 2 每一结点都有一状态 status , 以标记该结点当前是否可信。 $\text{status} = \text{In}$, 相信为真; $\text{status} = \text{Out}$, 不相信为真。此外, 每个结点都对应一个证实表 JL , 其中每一证实 J 形式为; $(\text{Inlist}, \text{Outlist})$ 。它为有效, 仅当 Inlist 表为空或当 Inlist 中每一个结点的状态均为 In , 且 Outlist 表为空或者 Outlist 表中的每一个结点的状态均为 Out 。 In 结点是指那些当前至少有一个有效证实的结点, 反之为 Out 结点。

定义 3 如果一个结点的 Inlist 表或 Outlist 表不为空, 则此结点表示一个假设, 否则此结点表示一个前提。如果一个假设 h 的 Outlist 表为空, 则称此假设为单调假设, 否则称为非单调假设。

定义 4 系统中信条的当前可信集定义为:

$$S_{\sigma} = \{r_i | \text{status}(n_i) = \text{In}, n_i \in N\},$$

当前不可信集定义为:

$$S_{\sigma^c} = \{n_i | \text{status}(n_i) = \text{Out}, n_i \in N\}$$

定义 5 对某一结点 n , 设其状态 $\text{status}(n) = \text{In}$, 则必 $\exists J_m \in JL_n$, 使得 $\text{status}(n) = \text{In}$,

称 J_m 为结点 n 的支持证实。

定义 6 结点 n_i 的支持结点集 $S_{sm}(n_i)$ 定义如下：

如果 $n_i \in S_{cr}$ ，且 n_i 的支持证实为 p_i ，则 $S_{sm}(n_i) = \{n_j | n_j \in Inlist(p_i) \text{ 或 } n_j \in Outlist(p_i)\}$ 。如果 $n_i \in S_{ncr}$ ，且 $JL_{ni} = \{p_1, \dots, p_k\}$ ，则 $S_{sm}(n_i) = \{n_j | p_j \in JL_{ni}, n_j \in Inlist(p_j) \text{ 且 } status(n_j) = Out \text{ 或 } n_j \in Outlist(p_j) \text{ 且 } status(n_j) = In, j=1, 2, \dots, k\}$

定义 7 结点 n 的支持结点关系为：

$$R_{sm}(n) = \{ \langle n, n_j \rangle | n_j \in S_{sm}(n) \}$$

如果 $N = \{n_1, n_2, \dots, n_k\}$ ，则系统支持结点关系为： $R_{sm}(N) = \bigcup_{i=1}^k R_{sm}(n_i)$ ，系统支持结点关系的传递闭包为： $R_{sm}^+(N) = \bigcup_{i=1}^k R_{sm}^+(N)$ 。我们定义结点 n 的祖先集为： $S_{ancest}(n) = \{n_j | \langle n, n_j \rangle \in R_{sm}^+(N)\}$ 。

定义 8 结点 n 的推论集定义为：

$$S_{con}(n) = \{n_k | \exists p_j \in JL_{n_i}, \text{ 且 } n \in Inlist(p_j) \text{ 或 } n \in Outlist(p_j)\}$$

其推论关系为： $R_{con}(n) = \{ \langle n, n_j \rangle | n_j \in S_{con}(n) \}$

设 $N = \{n_1, \dots, n_k\}$ ，则系统的推论关系为：

$$R_{con}(N) = \bigcup_{i=1}^k R_{con}(n_i)$$

定义 9 设 $N = \{n_1, n_2, \dots, n_k\}$ ，系统推论关系的传递闭包定义为： $R_{con}^+(N) = \bigcup_{i=1}^k R_{con}^+(N)$ 。定义结点 n 的反应集为： $S_{reply}(n) = \{n_j | \langle n, n_j \rangle \in R_{con}^+(N)\}$ 。

约定 1 冗余的知识有下面两种情况：

- (1) 两条一模一样的事实或两条一模一样的规则；
- (2) 两条规则之间有包含性关系如：

$$R_1: e_1 \wedge e_2 \wedge \dots \wedge e_m \rightarrow e$$

$R_2: e'_1 \wedge e'_2 \wedge \dots \wedge e'_n \rightarrow e$ 。当 $e_i = e'_i$ 且 $m \neq n$ 时，如果 $m < n$ ，则 R_2 为一条冗余的规则。

约定 2 矛盾的规则有下述两种情况：

- (1) 条件相同，但结论矛盾，如：

$$R_1: a_1 \wedge a_2 \wedge \dots \wedge a_n \rightarrow a \text{ 与}$$

$$R_2: a_1 \wedge a_2 \wedge \dots \wedge a_n \rightarrow \neg a \text{ 相矛盾}$$

- (2) 条件部分除含有互为矛盾的条件子句外的其余部分以及结论部分均相同，如：

$$R_1: a_1 \wedge a_2 \wedge \dots \wedge a_r \wedge a \rightarrow b \text{ 与}$$

$$R_2: a_1 \wedge a_2 \wedge \dots \wedge a_r \wedge \neg a \rightarrow b \text{ 矛盾}$$

约定 3 环路规则：规则中的事实可循环导出，如：

$$R_1: a_1 \wedge a_2 \wedge \dots \wedge a_m \rightarrow a'$$

$$R_2: a' \wedge a'_2 \wedge \dots \wedge a'_m \rightarrow a'$$

$$R_3: a'' \wedge a_2' \wedge \dots \wedge a_m' \rightarrow a_1$$

2 系统主要设计思想及运行实例

在我们所研制的知识库推理和维护系统中，对应结点只有简单命题（事实表示为 $p(X_1, \dots, X_n)$ ），规则 $P \leftarrow Q_1, \dots, Q_m$ ，转换为 rule (P, Q_1, \dots, Q_m) ，且每一结点都有一状态，以标记该命题当前是否为可信。这样不同于任何单调系统，在任何时候系统中存在可信命题也存在不可信命题。系统中命题的状态是随时间而不断变化的，新的事实、规则的加入可能使以前为可信的命题变为不可信，那么一切基于这个已变化命题的命题都要受到影响，需要得到修正。

系统所实现的自动推理和维护过程的主要步骤如下：

(1) 把命题 A 连同它的证实 J_A 加入知识库时，首先检查知识库中是否有此命题，如有，则转 1a，否则，把命题 A 加入知识库中。

1a. 判断证实 J_A 的 Inlist(J_A) 和 Outlist(J_A) 表中的任一结点 β 是否被结点 A 直接或间接证实。如果 $A \in \text{Sancent}(\beta)$ 或 $\beta \in \text{Sreply}(A)$ ，则表示结点 A 和结点 β 构成了一个循环证实回路，因而在本系统中只简单地把 β 从 J_A 中删除，以避免循环回路的出现。设通过上述检查后的 J_A 变为 J_A' 。

1b. 把 A 的证实 J_A 加入知识库中 A 的证实表 JL_A 中。

(2) 判断命题 A 在当前是否为可信命题，如不可信，则本次调用结束。

3a. 当命题 A 为可信时，则在知识库中进行推理，看是否引起矛盾，如引起矛盾，则调用 4 的相关性回溯过程，通过跟踪矛盾结点理由充足的证实，去掉引起该矛盾的非单调假设之一以消除矛盾并同时生成一个记录来避免以后出现类似的矛盾。

3b. 由命题 A 和知识库中的其它命题 β_1, \dots, β_m 进行推理，如果能推出其它的命题 C，则把命题 A 和命题 β_1, \dots, β_m 组成命题 C 的一个证实连同命题 C 一起转第 1 步，否则结束。

4a. 从产生矛盾的结点开始，回溯跟踪该矛盾的理由充足的支持，以便寻找导致矛盾的非单调假设集。具体做法是从矛盾起通过支持矛盾的状态为 In 的结点向后标记，直到找到一个非单调假设并返回所找到的非单调假设里的所有新假设。所谓新假设是不通过中间非单调假设直接支持矛盾的非单调假设。只修改新假设以避免在知识库中作很大的不必要的变动。

4b. 设返回的新假设表 $S = (A_1, \dots, A_n)$ ，从 S 中选取一假设 A_j ，把此假设的 Out 证实之一变为 In 证实。具体做法是：假设 A_j ，其证实 J_{A_j} 为： $(\text{Inlist}(\beta_1, \dots, \beta_j, \dots, \beta_m))$ ；再寻找一个新证实 J_b （可能为空），它由所有这样的命题组成，这些命题最终支持矛盾，但它们既不是 S 中的元素，也不为 S 中的元素所证实，设 J_b 为： $((A_1', \dots, A_i'), (\beta_1', \dots, \beta_j'))$ 。给命题 β_j 一个新的证实 $J_{\beta_j}((A_1', \dots, A_i', A_1', \dots, A_{j-1}', A_{j+1}', \dots, A_n'), (\beta_1, \dots, \beta_{j-1}, \beta_{j+1}, \dots, \beta_m))$ ，从而使 A_j 状态变为 Out 来消除矛盾，并把命题 β_j 和其证实 J_{β_j} 一起转入第一步，导致知识库的进一步和谐和完善。

下面举例来看看系统是如何进行推理维护的。

设系统中知识库所存的知识如图 2 所示：图中“-”表示对应的知识当前为不可信的，

“+”表示对应的知识当前为可信的。

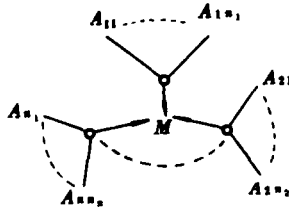


图 1

图 1 的含义为:知识 M 或由知识 A_{11}, \dots, A_{1n} 推出,或由知识 A_{21}, \dots, A_{2n} 推出,……,或由知识 A_{n1}, \dots, A_{nn} 推出。

hair(yang, yellow), eye(yang, blue),
 hair(zh, yellow), eye(zh, brown),
 contradiction ← is(X, one), is(X, two),

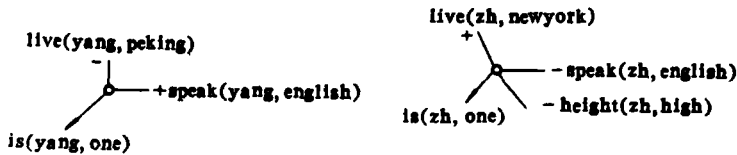


图 2

当把下述知识

$$\text{hair}(X, \text{yellow}), \text{eye}(X, \text{blue}) \rightarrow \text{is}(X, \text{one}) \quad (5)$$

$$\text{hair}(X, \text{yellow}), \text{eye}(X, \text{brown}) \rightarrow \text{is}(X, \text{two}) \quad (6)$$

加入知识库(图 2)中,看知识库的变化情况。

把知识(5)加入图 2 的知识库中,可推出 $\text{is}(\text{yang}, \text{one})$ 这个新知识,但由于知识库中已存在 $\text{is}(\text{yang}, \text{one})$ 这条知识,为保证知识库的非冗余性,则只把关于 $\text{is}(\text{yang}, \text{one})$ 的证实加入其证实表中。如图 3 所示加入知识(5)之后的知识库。

把知识(6)加入图 3 所示的知识库中,可推出新的知识 $\text{is}(\text{zh}, \text{two})$,使得知识库因同时存在 $\text{is}(\text{zh}, \text{one}), \text{is}(\text{zh}, \text{two})$ 这两条相互矛盾的知识而导致知识库的不一致性。当系统感知矛盾后,则沿着推理链逆向追踪,找到支持产生矛盾的有关假设(标有“-”号)。修改为其相反的状态,即修改为“+”。然后再沿正向推理链修改所做的假设以消除矛盾。完整的做法是通过给知识 $\text{speak}(\text{zh}, \text{english})$ 一个新的证实: $[[\text{is}(\text{zh}, \text{two}), \text{contradiction} \leftarrow \text{is}(X, \text{one}), \text{is}(X, \text{two})], [\text{height}(\text{zh}, \text{high})]]$,使得 $\text{speak}(\text{zh}, \text{english})$ 这条知识在当前知识库中变为可信的,从而使 $\text{is}(\text{zh}, \text{one})$ 变为不可信而消除知识库的不一致性。最后的知识库如图 4 所示。

hair (yang, yellow), eye (yang, blue),
 hair (zh, yellow), eye (zh, brown),
 contradiction ← is (X, one), is (X, two),
 is (X, one) ← hair (yellow), eye (X, blue),

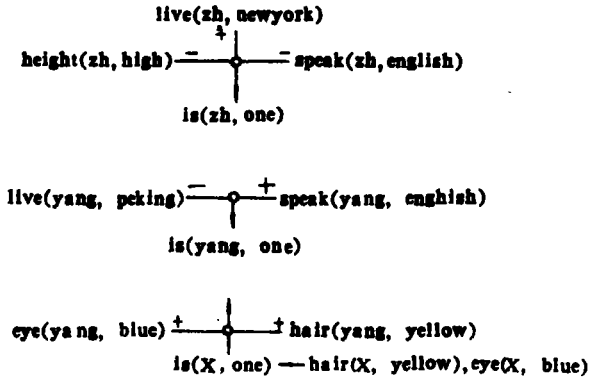


图 3

hair (yang, yellow), eye (yang, blue),
 hair (zh, yellow), eye (zh, brown),
 contradiction ← is (X, one), is (X, two),
 is (X, one) ← hair (X, yellow), eye (X, blue),
 is (X, two) ← hair (X, yellow), eye (X, brown),

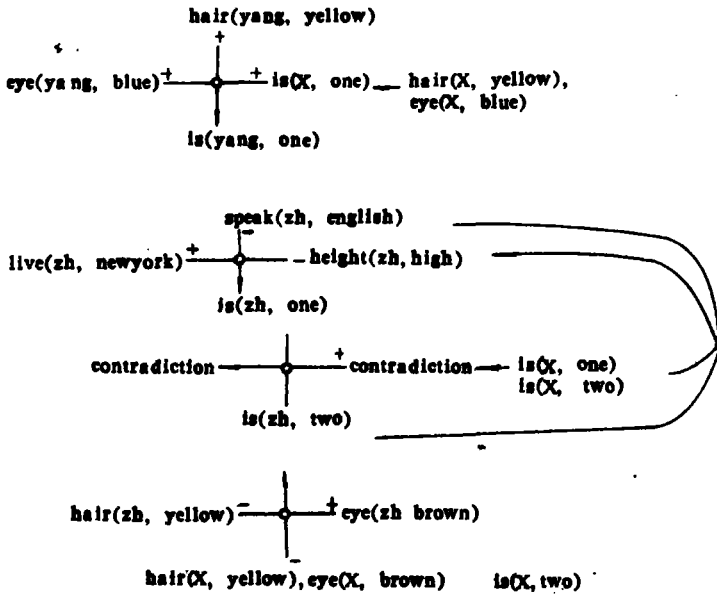


图 4

3 结束语

知识库推理和维护系统 KBIMS 作为知识获取系统 GKD—NKAS 的一个子系统已在 VAX—II/780 上实现。此系统利用非单调逻辑及其推理理论中缺言推理的理论,对知识库中的知识进行推理维护。它承认不在系统之内的命题不一定是假命题,仍然可能是真命题,并且随着知识的增加原来系统内部的命题也可能变为假命题。系统随着时间的推移可能不是增大而是减小,使系统的性能得到改善。由此可见,用此系统进行知识库的自动推理和维护可以使知识库更加智能化,并且可减轻系统开发人员的负担,提高智能系统的有效性和实用性。

致 谢

本课题的研究过程中,吴泉源教授、胡运发副教授提出过很多有益的意见和建议,在此向他们表示感谢。

参 考 文 献

- [1] Davis M. The Mathematics of Non—monotonic Reasoning. *Artif. Intell.*, 1980, (13)
- [2] Reiter R. A Logic for default Reasoning. *AI, April*, 1980, 13 (1—2)
- [3] Doyle J. The Ins and Outs of reason maintenance. 1983, *IJCAI—8*

Knowledge Base Inference and Maintenance System (KBIMS)

Yang Li Hu Shouren

(Department of computer Science)

Abstract

How to make the knowledge base have the functions of automatical inference and maintenance has been a key problem existing in the design of the knowledge base that needs to be practical and effective. We propose an automatical knowledge base inference and maintenance method based on default theory, and then we introduce a knowledge base inference and maintenance system, which has been implemented on VAX-11/780 machine in GKD-Prolog.

Key words artificial intelligence, knowledge base, non-monotonic inference, default theory, Prolog, knowledge base inference and maintenance