知識獲得システムの一実現法

北上始，宮地泰造，国藏進，古川康一
(財) 新世代コンピュータ技術開発機構

1、はじめに

人間の知的な精神活動を計算機システムに行わせるためには、種々のメカニズムを備えなければならないが、その一つとして、人間がもつ知識を恒常的に蓄積管理する機構があげられる。これは、知識ベースシステムの研究として知られている。

我々は、この研究の一環として、知識獲得システムの研究を進めている。知識獲得とは、知識ベースに対する知識の同化及び調節などによる知識の収集を意味する。

以下では、知識ベースに蓄積されている知識を、(i)ファクト（個々の事実）、(ii)ルール（一般的な規則）、(iii)IC（ファクトとルールについての矛盾性制約）、としてとらえる。

本稿では、知識ベースシステム向きの知識獲得システムについて、その概念構成及び一実現法を述べる。本システムのインプリメントに際しては、計算機システムDEC2060上で作動するProlog－10を利用した。

2、知識獲得システムの概念構成

図1は、知識ベースシステム向きの知識獲得システムの概念構成図である。

図中のメタ推論機能としては、命題の証明可能性を判定できるメタ述語を利用している。メタ述語は、次の2種に大別される。1つは、存在限定子だけからなる命題を証明するメタ述語（existential-demo述語）であり、他の1つは、全称限定子だけからなる命題を証明するメタ述語（universal-demo述語）である[1,2]。これらにより、知識獲得の際に生じる各種の定理証明問題を解くことができる。

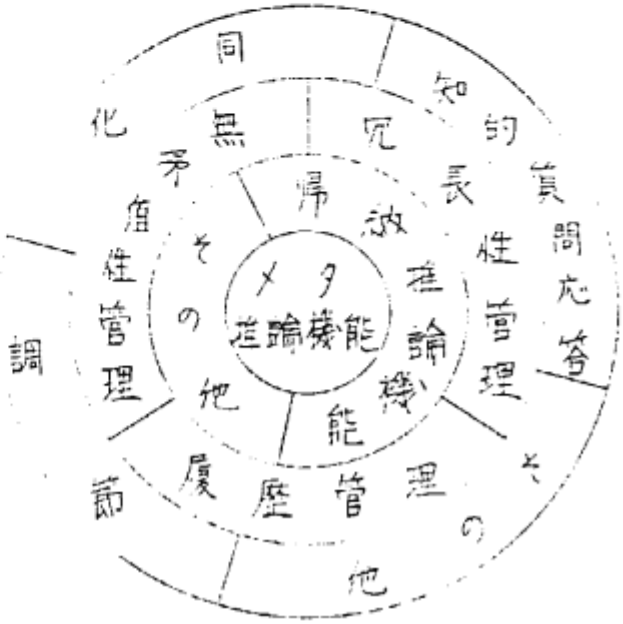

図1. 知識獲得システムの概念構成

帰納推論機能は、Shapiroの研究によるモデル推論を採用している[3]。知識獲得における本機能の役割としては、次の項目があげられる。

(1)ユーザの意図に合った知識の作成
(2)誤り知識（主にルール，IC）の修正
(3)誤り知識の管理による一無化

現在、(1)，(2)については、ある程度まで解決しており、(3)については、今後の課題としている。

冗長性管理は、知識を獲得した際に、知識ベースが冗長な知識をもつことをさけたいと思うユーザのために用意されている。

履歴管理は、知識ベースにする質問

せ、調節のプロセスを記録することにより、現在、過去、未来の知識に関する各種の論証をサポートすることができ、現在、この方式を検討中である[4]。

無矛盾性管理は、知識ベースの論理的矛盾が発生するのを防止する機能である。

同化は、ユーザの意図に合った知識を蓄積するために利用される。ユーザの意図は、IC として定義される。この基本的なアイデァは、Kowalski に基づいている[5,1]。

一方、調節は、外界の知識を常に正しいと見做し、外界（ユーザ）の入力知識に合わせて、知識ベースを修正するために利用される。調節の方法には、
(1)ユーザがファクトを与える方法、
(2)ユーザが直接知識を置換する方法、
(3)（1）と（2）を組み合わせる方法、
があげられる。

知的質問応答は、知識ベースの状態を、質問応答により確認するために利用される。ここでは、高度な知的インタフェース機能が要求される。

以上の機能により、ユーザは、知識ベースの状態を監視しながら、知識ベースの同化と調節を繰り返し、知識ベースを、ある目標へと注意深く成長させていく事ができる。

3．メタ推論

前述の2種の demo 述語を実現するために共通に使われる demo 述語について述べる。この demo 述語は、あるゴールの証明可能性だけを調べる述語であり、真偽値として 0（'true'），2（'false'）を返す能力をもっている。どちらの値も発見できなかったとき、demo 述語を fail させている。

これにより、ゴールの証明結果として、'false' 値をとった場合、ゴールの証明プロセスを検討して、その原因となる知識を判断することができる。図2に、demo 述語の実現例を示す。ここでは、次の前提を置いている。

(1) 知識ベースには、正の知識と負の知識があり、負の知識は、互いに独立である。

(2) 負の知識の形式は、not((Head:-Goals)) であり、Head自身は、not(・) の形式をとらない。

(3) 知識ベースの個々の知識には、ID が付けられており、ユニークに識別できる。

(4) ゴールには、変数が存在しない。

（universal_demo 述語を実現するときは、判定となる部分をスコーレム化する方式を採用しているので、この制限は、問題にならない。知識ベースへの質問応答のために使用される demo 述語は、Prolog インタプリタと呼ばれる述語を使えば良い[1,2]）

```
demo(KBL,0,CNTL,0,[]):-! .
demo(KBL,2,CNTL,2,[]):-! .
demo(KBL,(P;Q),CNTL,R,St):-! ,
   (demo(KBL,P,CNTL,R1,St1);R1=1) ,
   (R1=0 ->R=0 ;
    (demo(KBL,Q,CNTL,R2,St2) ;R2=1) ,
    (R2=0 ->R=0 ;R1=2,R2=2 ->R=2 ;fail)) .
demo(KBL,(P,Q),CNTL,R,St):-!,
   (demo(KBL,P,CNTL,R1,St1) ;R1=1) ,
   (demo(KBL,Q,CNTL,R2,St2) ;R2=1) ,
   R3 is R1+R2 ,
   (R3=0 ->R=0 ;
    R3>=2,R1=\=1 ->R=2 ;fail) ,
   append(St1,St2,St) .
demo(KBL,not(Goals),CNTL,R,St):-! ,
   demo(KBL,Goals,CNTL,R1,St) ,
   (R1=0 ->R=2 ;R=0) .
demo(KBL,Head,CNTL,0,[[ID,(Head:-Goals)]|St1]):-
   setof([ID,Goals,St1],
    (clause_m(KBL,Head,ID,Goals,CNTL),
     demo(KBL,Goals,CNTL,0,St1)),S1),! ,
   member([ID,Goals,St1],S1) .
demo(KBL,Head,CNTL,R,[[ID,not((Head:-Goals))]|St2]):-
   setof([ID,Goals,St2,R2],
    (clause_m(KBL,Head,ID,Goals,CNTL),
     demo(KBL,Goals,CNTL,R2,St2)),S2),! ,
   member([ID,Goals,St2,R2],S2),R2=0 ->R=2 .
```

図2．demo 述語の実現例

4．おわりに

現在、帰納推論の高速化について検討を進めており、その成果を得ている。また、Belief System としても有効な役割が果せるように研究を進めている。

【参考文献】 [1] 北上 他：推論知識獲得と人工知能研究会資料 33-2 '83.6，[2] 国藤 他：[illegible]
[3] Shapiro,E.Y.：Algorithmic Program Debugging, Yale Univ. Research Report '82，
[4] Doyle,J.：Truth Maintenance System for Problem Solving, AI-TR-419 '78，
[5] Kowalski,R.A.：Logic as a Database Language, Imperial College '81 .