知識工学と人工知能　41－6
（１９８５．７．４）

知識ベース管理システム（ＫＢＭＳ）

服部　文夫　清水　信昭　土屋　秀幸　桑原　和宏　和佐野哲男
（ＮＴＴ　横須賀電気通信研究所）

１．はじめに

近年の知能処理応用システムの開発意欲の高まりに対応して、数多くのエキスパートシステム構築用ツールが開発されつつある[1～3]。これらのツールは1970年代の実験的なツールに比べて、知識表現能力の向上、支援系の充実等の面で強力なものとなってきている。

しかし、これらのツールを現実のアプリケーションシステム構築に適用しようとする場合、まだ十分に満足できるものとは言えない。第一は知識の表現能力の問題である。最近のツールの知識表現はほとんどがフレームにもとづいている。フレームによる知識表現は確かに柔軟ではあるが、知識を体系化するという面では基本的にはis-a階層体系だけである。したがってその他の体系化を行なうには、利用者が暗にそれを意識して処理手続きを記述することになり、利用者の負担を増加させている。

第二は実用性で、プロトタイピングが終了して実働システムに移行しようとした場合に、性能的に十分でない、あるいは動作環境が専用のＷＳに限定される等の問題がある。

そこで、我々はこれらの問題を解決し、より実用的なエキスパートシステム構築ツールを目指して、『知識ベース管理システム（ＫＢＭＳ）』の開発を進めている。本システムの具体的な狙いは次の３点である。

(1) 多様な応用システムを容易に構築可能とするための、強力な知識表現能力の提供。
(2) 実働環境としての現実的な性能。
(3) 特別なＷＳ等の環境だけでなく、汎用機等の実用的な環境での動作。

本資料ではこれらの狙いを実現するための本システムの設計方針とその実現方法について述べる。

２．設計方針と構成概要

上記の狙いに対応する具体的な設計方針と、システム構成の概要について述べる。

２．１　設計方針

(1) 知識構造の表現能力の強化

知識を効率よく格納し、有効に利用するためには、知識を体系化することが重要である。体系化の方法はフレーム表現で通常提供されているis-a階層による体系化の他にも、part-of 関係による体系化や、あるいはアプリケーション固有の意味にもとづく体系化も考えられる。これらの体系化をサポートする機能をシステムで提供することで、知識の記述が容易になり、かつその利用も効率よく行なうことができる。本システムでは知識の体系化を支援するため、従来のフレーム知識表現に対して知識構造の表現機能を強化している。

(2) 推論処理の高速化

エキスパートシステムの性能を考えた場合、基本的な知識のアクセス速度の向上も勿論必要であるが、最もネックになるのは推論の実行時間である。そこで本システムではその高速化をはかるため、推論実行方式の一つにプロダクションシステムの高速な実行方式として定評のあるRETEアルゴリズム[4] を採用し、さらにその改良をはかることとした。

(3) 汎用機とパソコンの機能分担

先にも述べたように、実用的な動作環境として汎用機上での動作を可能とする。しかし、汎用機に単に端末を接続しただけでは、いわゆる高機能WSを利用した場合のようなマルチウィンドウやマウスによる良好なインタフェースの実現は困難である。そこで端末としてパソコンを用い、知識ベース作成支援系の機能の大部分をパソコンに分担させることで、マンマシンインタフェースの向上と実用的な実行環境の両立をはかっている。

2．2　システムの構成概要

本システムの構成概要を図1に示す。

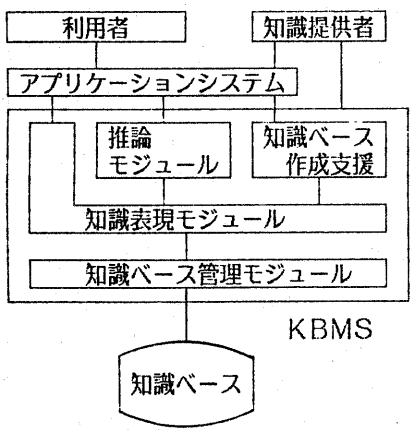

図1　知識ベース管理システムの構成

①知識表現モジュール

知識をフレーム、ルール形式で表現し、これをアクセスするモジュール。ただし、ルールも実際にはフレームとして管理される。

②推論モジュール

ルール知識をもとに推論を行なって問題解決を行なうモジュール。

③知識ベース作成支援モジュール

知識提供者の知識投入を支援するエディタ、デバッガなど。

④知識ベース管理モジュール

知識ベースの管理を行なうモジュール。二次記憶との記憶階層管理や、世代管理、共用管理等を行なう。

3．知識表現

3．1　知識表現の基本的考え方

本システムの知識表現はフレーム表現をベースとし、これにルール表現を組合せた統合表現である。これは知識の種類に応じて様々な知識表現法を組合せる必要があるという、最近のツールに共通する考え方にもとづいている。

ところで、知識の意味はそれが他とどういう関係にあるかで表現されるものであり、また知識を利用する上でも知識相互が関係付け、体系化されていること（これを知識の構造化と呼ぶことにする）が重要である。しかし、従来のフレームベースの表現法では、フレームで表現される概念の間の関係として陽に表現されるのはis-a関係だけであり、他の関係はスロットの値として他のフレームへのポインタを埋め込むことで表現されていた。したがってアプリケーション対応に知識を構造化し、その意味解釈を定義するには利用者がその手続きを一から記述する必要がある。これは利用者にとって大きな負担であると同時に、システムによる実行の最適化や支援系でのサポートが限定される等の問題がある。そこで本システムではフレーム知識表現に知識の構造化のための機能を強化し、それに対する操作機能をシステムでサポートすることで、これらの問題の解決をはかっている。

3．2　知識構造の表現

(1) 知識の構造化

知識の構造化とは知識をまとめたり、関係づけることで、その意味を明確化するとともに、操作を容易化するものである。

構造化は以下の２つに分類できる。

①問題に依存しない、どのような知識にも共通的に行なわれる構造化。主として知識の抽象化による構造化である。

②問題対応に定義される構造化。

本システムでは前者に対する機能をシステムで標準的に提供し、後者については利用者による定義を容易化する枠組を提供している。

(2) システム定義の構造化表現

人間が知識を構造化する際に、共通的にとられる手法として以下が考えられる[5]。

①classification

instance-of 関係。同じ属性を持つフレームの集合をクラスとして抽象化する。

②generalization（汎化）

is-a関係。クラスＡのインスタンスがすべてクラスＢのインスタンスである時、ＢをＡの汎化であるという。

③aggregation （集約化）

part-of 関係。フレームＧがいくつかの異なる種類のフレームＨ～Ｋの集合で構成される時、ＧはＨ～Ｋの集約であるという。

④association

member-of 関係。同じ種類のフレームの集合を一つのフレームとしてとらえる。

本システムではこれらの知識構造の表現方法をシステムとしてサポートしている。①はクラス・インスタンスの関係、また②はスーパークラス・サブクラスの関係としてサポートしている。③はグループとエレメントの関係として、インスタンス間に関係づけを可能とするとともに、それを抽象化してクラス間に関係を定義することも可能としている。④はグループとエレメントの関係の特殊な場合として表現する。（図２参照）

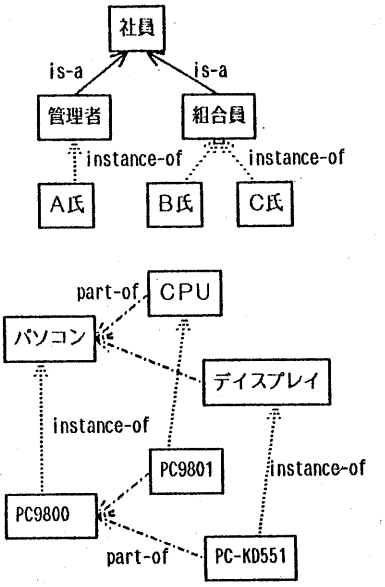

図２　システム定義の構造化表現の例

(3) 利用者定義の構造化

利用者定義の構造化とは、概念間に特定の意味を持つ関係を付けること、例えば「夫」と「妻」の間に「夫婦」という関係を付けることを言う。このような関係は応用に固有のものであり、ツールとしては一般的な関係を定義できる枠組を提供することが必要である。この枠組の表現法としては次の２点が要求される。

①関係付けができると同時に、関係自体の構造化が可能なこと。

②関係に対応する操作を記述できること。

後述するように、構造に対する操作（解釈手続き）が定義されて始めて構造が意味を持つ。

上記を満たすために、本システムでは関係自体をフレームとして表現することとした。このフレームを関連フレームと呼び、インスタンス間の関係を表現する関連インスタ

ンスと、そのクラス表現である関連クラスとに分けられる（図3参照）。関係をフレームとすることで、関係自体のgeneralizationやaggregation 等も自然な形で表現可能となる。また関係に対する操作も関連フレームに付加された手続きによって表現できる。

表1　構造に対する操作機能（一部）

操作名	機能概要
GET	スロットを参照する。必要な場合にはis-a階層をたどる。
PUT	スロットを更新する。必要な場合にはis-a階層をたどる。
INSTANCE	指定クラス配下で指定条件を満たすインスタンスを検索。
NEWGROUP	グループおよびその配下のエレメントを作成する。
DESTROYGROUP	グループおよびその配下のエレメントを消去する。
GETGROUP	指定エレメントを配下とするグループを得る。
GETELEM	指定グループの配下のエレメントを検索する。
NEWREL	指定されたフレーム間を関係づける関連フレームを作成する。
GETENT	指定された関連フレームと関係づけられているフレームを得る。
GETRENT	指定されたフレームと指定された関連関連で関係づけられているフレームを得る。
CLASS	指定されたインスタンスフレームのクラスを得る。
REPLACESUPERS	指定フレームのスーパクラスを変更する。

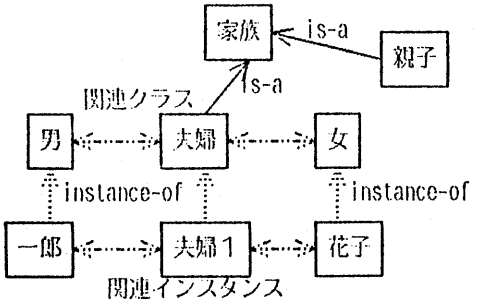

図3　利用者定義の関係の表現例

3．3　構造に対する操作

構造に対する操作とは構造をその意味に応じて解釈し、知識を利用する手続きであり、フレームに付加される手続きによって実現される。前節で述べたシステム定義の構造に対しては、システムでその意味に応じた操作機能が提供されている。例えば、classificationに対してそのインスタンスを検索する手続き、aggregation に対してグループを表わすフレームを生成すると同時にそのエレメントを合わせて生成する手続きなどである。またgeneralizationに対応してスロットのアクセス時に属性継承が行なわれる。本システムで提供している主な操作機能を表1に示す。利用者はこれらの基本的な操作機能を用いて固有の意味付けを構築することができる。現在サポートしている操作機能は十分ではないが、これを強化していくことで利用者の負担が低減され、さらに構造からくる制約のチェックによる知識ベースの一貫性の保持、あるいは実行の効率化等が可能になってくる。

3．4　フレーム知識表現の実現

(1) フレーム構成

本システムのフレーム構成を図4に示す。variableは値を持つスロット、methodは手続きを格納するスロットである。instance variable 、instance method はクラスのみにあるスロットで、前者はインスタンスのvariableの型、後者はインスタンスに共通のメソッドをクラスに記述するものである。instance variable はインスタンスに継承されるとvariableになる。

(2) 属性継承の実現

本システムでは多重継承（マルチプルインヘリタンス）を可能としている。すなわち、複数のスーパフレームからの属性継承ができる。また、属性継承の制約をスーパクラス側から指定できる。

さらに属性継承にともなう性能劣化を避けるため、継承されているスロットへの最初のアクセス時に継承元へのポインタがは

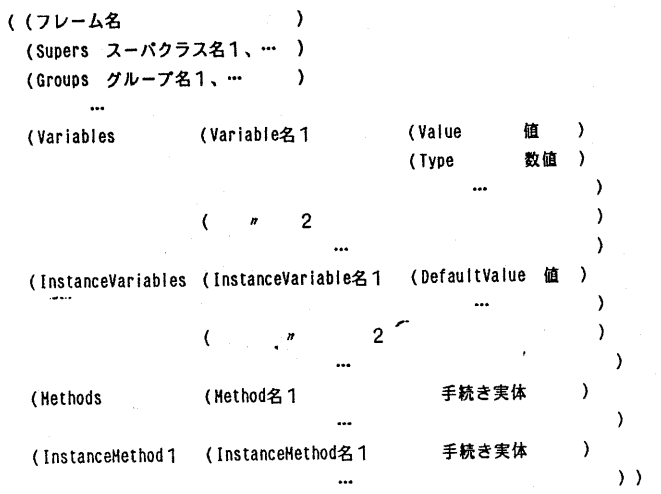

図4 フレームの形式

られる。

4．推論の高速化

本システムの推論メカニズムは、フレームで表現された知識をワーキングメモリ(WM)とし、これとルールを照合して推論を実行するプロダクションシステムである。推論の実行方式としては、アプリケーションに応じて多様なものを提供しているが、特に実用的な性能という観点からその1つとして高速な前向き推論方式を提供している。

4．1 パターン照合処理の効率化

プロダクションシステムの実行はパターン照合、競合解決、実行のサイクルを繰返すことによって行なわれるが、実行時間の大部分を占めるのはパターン照合である。その高速化手法としてReteアルゴリズム[4]を採用し、更にその改良を図った。

（1）Reteアルゴリズム

ReteアルゴリズムはOPS5で採用されている推論実行方式で、以下の考えに基づいて、条件照合処理の効率化を図っている。

①ルールのコンパイル時に全ルールの条件を基本的な条件テストに分解して推論ネットワークを作成し、ルール間に同一な条件がある場合は条件テストを共通化する。

②以前に実行した条件照合の途中結果をネットワーク中に保持しておき、ルールの実行中にインスタンスの更新があった時には、更新のあったインスタンスに関してのみ再照合を行なう。

（2）構造を利用した効率化

Reteアルゴリズムにおける条件照合処理は1つのクラス内に閉じた条件照合と、複数クラスにまたがる条件照合の2段階からなる。このうち複数クラス間の照合が実行時間の大半を占めている。本システムではクラス間に関係が陽に表現されている場合には、照合処理の代わりに関係をたどることによって高速化した。例えば『variable"A1"の値が"XX"であるクラスA のインスタンスと関連を持つクラスB のインスタンスが存在するか』という条件について考える。陽な関係表現がない場合は、クラスA のvariable"A1"の値が"XX"であるインスタンス及び、クラスB のすべてのインスタンスをメモリノードに蓄え、それぞれに蓄えられたすべてのインスタンスの組合わせについて照合処理を行なう必要がある。これに対し、構造が陽に表現されている場合、クラスA とクラスB のインスタンスは直接に関係づけられているので、インスタンス間の照合処理は、クラスA のインスタンスから、クラスB のインスタンスへ関係をたどる処理となり、不必要な照合処理を回避できる。

（3）ルールの記述

前節で述べた高速化を実現するために、

本システムでは、フレーム間の関係に関する条件を陽に記述可能としている。例えば、『CLASS-A とCLASS-B の間にグループという関係があるか』という条件は
IF CLASS-A _@GROUP_CLASS-B ～
のように記述する。

また、THEN部においては、フレームの値の更新の他に関係づけの変更等関係操作用のアクションの記述が可能である。これにより、ルールの記述も容易となる。

4．2 実現方式

(1) 関係ノードの導入

関係を利用して推論を実現するためには、条件照合処理時に関係をたどる処理が必要である。関係をたどる契機として
①条件テストを行なう前に入力されたインスタンスにつながるすべての関係をたどる
②条件テストを行ないながら必要な時点で動的に関係をたどる
という2つが考えられる。①がすべての関係をたどるのに対し、②では途中の条件テストを通過したインスタンスについてのみ関係をたどる処理を行なえば良いことから②の方法をとることとする。これを実現するために、推論ネットワークの構成要素として関係をたどる処理を行なう関係ノードをあらたに導入する。

図5に従来の推論ネットワークと、関係を導入した時の推論ネットワークを示す。インスタンスが途中の条件テストを通過し、関係ノードに達すると、そのインスタンスと関係でつながっているインスタンスを見い出し、関係するインタンスをリストにしてネットワークの下方へ送りだす。

(2)WM 更新時の照合処理

Reteアルゴリズムでは、メモリノードに条件にマッチしたインスタンスを保持して

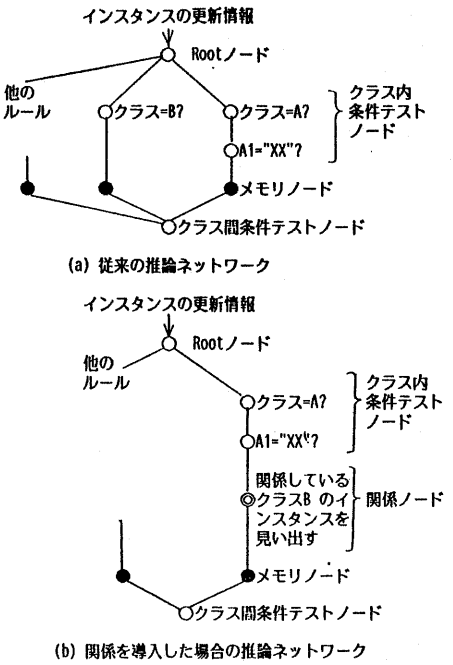

図5 推論ネットワークの例

おき、ルールの実行中に更新のあったフレームについてのみ再度照合処理を行なう。関係を利用した推論では、関係の有無による条件もあるので、たどった関係についてもその情報をネットワーク中に保持しておき、更新時には関係を先頭からたどることなく効率的な処理を行なっている。

4．3 方式の評価

関係表現機能を導入することによる処理時間短縮の効果をデータベース性能診断システム[6] をモデルとして行なった。図6に推論対象のインスタンスの数と、関係導入による効果の割合を示す。

5．パソコン上での知識ベース作成支援系

5．1 汎用機とパソコンとの機能分担

本システムは、実用的な動作環境を提供するため、ホストとして汎用機を使用して

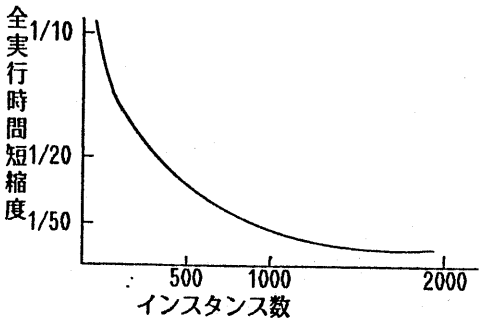

図６　関係機能導入による実行時間の短縮度

いる。しかし、端末として単なるキャラクタディスプレイを接続しただけではマルチウィンドゥ等の良好なマンマシンインタフェースを実現できない。そこで端末としてパソコンを用い、ホストとの機能分担を図ることでこの問題を解決している。この際、端末側には単なるコマンド入出力や画面制御だけでなく、ホストに依存しない機能（エディット機能の大部分）を極力端末側に持たせることで、応答性の向上を図っている。この実現のために、パソコン上にホストから送られてきたフレーム等を保持している。（図７参照）

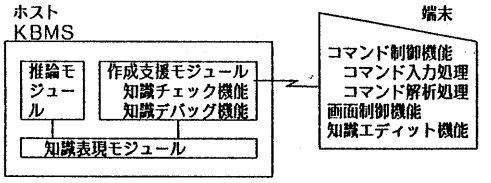

図７　ホストと端末の機能概要

ユーザインタフェースとしては、マルチウィンドゥをサポートし、ルール作成時やフレームの編集時等に、かかわりのあるフレーム、関係表示等を同時に見ながら編集等が可能としている。また、マウスを利用したメニュ選択によりKBMSの各種操作を可能とすることで、熟練していないユーザにも、KBMSが容易に利用できるようにしている。画面イメージを図８に示す。

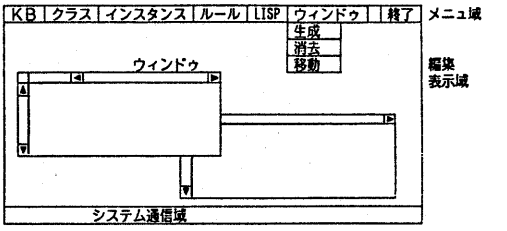

図８　画面イメージ

５．２　知識ベース作成支援機能

本節では、本システムで提供している具体的な作成支援機能について述べる。提供に当たっては、知識ベースを作成する立場から、①知識を様々な観点から効率良く整理でき、知識間の複雑な関係が視覚的に捕えられる②知識の編集が容易である③蓄えられた知識の参照、更新の状況が容易に把握できることに重点を置いている。

（１）フレーム知識の関係表示・関係の編集機能

複雑な構造を持つ知識間の関係を利用者にわかりやすい形で表示、あるいは編集させるための機能として以下の機能がある。

(a) is-a関係の表示
(b) instance-of 関係の表示
(c) part-of 関係の表示
(d) 関連の関係表示

is-aの表示例を図９に示す。

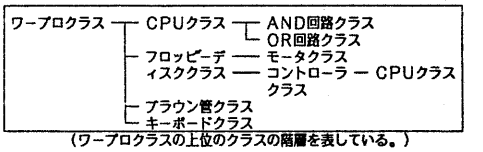

図９　is-a関係の表示例

（２）フレーム、ルール知識の表示、編集機能

フレーム、ルールそれぞれの構造に応じたスクリーンエディタを提供し、項目単位（例えば、variable、ルール文単位）、及び文字毎の編集を可能としている。また、知識ベースの初期投入時等では、パソコン上でローカルに知識の編集を行ない、後で一括してホスト上に知識ベースを作成することも可能としている。

（３）　知識のデバッグ機能

検索や推論の筋道のデバッグを効率良く行うためには、フレームやルール知識のトレースやブレークを行い、デバッグ情報を取れることが必要である。本システムでは表２に示すトレースの取得契機で、必要なトレース情報を取れるようにしている。

トレース情報は、指定によりファイルに出力され、エディタのサーチ機能により必要な情報の検索が可能である。また、トレースの他にブレークも可能であり、トレースと同様の契機で、その時のフレームのvariableの状態等が参照できる。

表２　知識のトレース

種別	トレースの取得契機	トレース情報
フレーム間のメッセージのトレース	指定されたフレームのメッセージ受信時	メッセージの内容と返却値
	指定されたクラスのインスタンスのメッセージ受信時	フレーム内の変数の値
	指定されたメソッド（セレクタ名）の送信時	
ルールのトレース	ルールの条件部の評価前	ワーキングメモリ、ルール内の変数の値
	ルールの実行部の実行前	
	ルールの実行部の実行後	

６．おわりに

知識ベース管理システムＫＢＭＳについて、知識構造表現を強化した知識表現、推論の高速化手法、汎用機とパソコンの分担による支援系の実現の３点について述べた。今後さらに改良を進めていく必要のある課題としては、

①システムでサポートしている知識構造に対する操作機能の強化、
②推論性能の一層の向上、
③支援系、特にデバグ機能の強化、ＷＳ環境での表示機能の向上、
などがあげられる。

ＫＢＭＳは既にDEC2060 上で稼働中であり、さらにDIPSやSymbolics 等への移植を進めている。現在ＫＢＭＳを使用して各種の応用システムを開発中であり、その評価を通してより一層の改良を進めていく予定である。

<謝辞>

本研究の機会を与えていただき、かつ日頃からご指導戴くＮＴＴ横須賀通研知識ベース研究室寺島室長に感謝いたします。

<参考文献>

[1] D.G.Bobrow and M.Stefic,The LOOPS Manual,Preliminary Version,Xerox Corp.,1983

[2] "IntelliCorp KEE Software Development System User's Manual,IntelliCorp,1985

[3] B.D.Clayton,"ART Programming Primer",Inference Corp.,1984

[4] Forgy,C.L.,"Rete:A Fast Algorithm for the Many Pattern Match Problem",Artificial Intelligence 19,1982

[5] M.L.Brodie,J.Mylopoulos,and J.W.Schmidt ed.,"On Comceptual Modelling",Springer-Velag,1984.

[6] 馬場他,"データベース性能診断システムの実現法"，情処全大 ,30