

TM-0189

PSI上のエキスパートシステム
開発支援ツール(1)～(4)

澤本 潤, 久保野秀雄,
永井保夫, 岩下安男

August, 1986

©1986, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

PSI 上のエキスパートシステム開発支援ツール (1) ～ 基本構想 ～

5K-1

澤本 直、久保野秀雄、永井保夫、岩下安男
((財) ICOT)

進藤 静一
((株) 三菱電機)

1. はじめに

エキスパート・システムは現在、人工知能の最も有望な応用分野の1つである。その有効性が特定の分野では理解されるようになり、さらに適用範囲の拡大が望まれている。しかし、現在みられるレベルのエキスパート・システムでは解決されるべき種々の問題がある [1]。我々は、そういった種々の問題を把握し、その解決を試みる為の第1歩としていわゆる第2世代レベルのエキスパート・システム開発支援ツール PROTON (Prototype Shell for the Next Generation) の開発を PSI 上で行っている。

種々の分野における応用問題をツール上で実際にインプリメントし、応用領域の分析 (たとえば、どのような知識表現や問題解決の手法が必要か等) を行うことによって、知識情報処理に必要とされる基礎的な技術を明確にすることが可能となる。本ツール上で種々の分野の応用を適用して問題点の認識を行うと言う意味で、特定の分野に特化した形式のものではなく、ある程度汎用性をそなえたツールが望ましい。

このツールは、FGCS (第五世代コンピュータ) プロジェクトの問題解決・推論、知識獲得、知識ベース管理、知的インタフェースなどの成果を適宜盛り込んで実験を試みる為の実験システムとしても利用する。

2. 基本構想

図1にPROTONの全体構成を示す。PSI 上のツールとして以下のような特徴を持っている。

- (1) ESP をベースに、オブジェクト指向とロジック・プログラミングを生かしたフレーム・ベース、ルール・ベースの実現。
- (2) PSI, SIMPOS の機能を活用した快適なユーザ・インタフェースの實現。
- (3) ユーザによる機能追加を容易にするオープンなシステム構成。
- (4) 各種データ収集機能。

(1) は複数の知識表現モデルを組み合わせたハイブリッド

な知識表現環境をESP のオブジェクト指向とロジック・プログラミングの枠組みで實現する。問題領域の構成要素に関する知識 (静的な知識) はフレーム・ベースとして表現し、問題解決のヒューリスティックスに関する知識表現としてルール・ベースを提供する。問題解決の戦略 (知識の使用) に関するメタな知識の表現としてメタ・ルールの記述が行なえる。

(2) のユーザ・インタフェース (I/F) としては、構築者 I/F と利用者 I/F の2つがある。構築者 I/F はマルチ・ウィンドウやメニューを使用した快適さをめざす。利用者 I/F については普通のシェルにみられるような説明機能、ユーザへの質問機能が構築容易なようなプリミティブを用意する [2]。

(3) は、基本的にESP のオブジェクト指向により、新しい機能の組み込みが容易におこなえると言うことである。たとえば、確信度付推論機能なども図2にみられるユーザ定義関数として追加できる。また、推論機構、利用者 I/F などユーザによるカスタマイズが容易なように配座されている。

(4) は、ルールやファクトへのアクセス頻度、システムの状態遷移などの実行時データを収集する機能である。目的としては、ツールもしくはそのベースとしてのマシン環境の評価とエキスパート・システムに蓄えられた応用領域の知識 (ルールやファクト) の性格を知るといった2つがある。

3. 機能概要

PROTONのルール・ベースとフレーム・ベースの戦略構成を図2に示す。これらの機能概要を以下で説明する。

3.1. フレーム・ベース [4]

問題領域の構成要素と構成要素間の関係定義がフレームとしてテンプレート (ESP の CLASS) 表現される。推論の過程でルールからの処理の対象となるのは、テンプレート表現から実体化され、作業領域 (Working Memory: WM) に格納されたインスタンスである。構成要素はフレームとしての通常の属性継承や属性値の操作に対するデモン

Expert System Shell on PSI (1) ~ Overview ~

J. SAWAHITO, H. KUBONO, Y. NAGAI, Y. IWASHITA, S. SHINDO (*)
ICOT, (*) Mitsubishi Electric Corporation

能を備えている。属性値が未定義の場合にユーザに質問したりユーザからの“WHY”に答える機能はシステム組み込みとした。

関係定義では構成要素のインスタンス間のユーザ定義によるなんらかの関係が表現される。関係に伴う制約条件チェックの付加手続きの記述を行う事によりある程度の意味記述が可能となっている。関係定義においても関係概念の一般性に基づく帰納的な、性質や制約条件の継承を考える。例えば、家族という関係は夫婦や兄弟などより、より一般的な関係である。また、システムとして逆関係や間接関係等の等価関係の探索機能を提供している。関係定義も1つのフレーム要素としているので多項関係の表現も容易である。

3.2. ルール・ベース [5]

ルール・ベースは図2に見られるように複数の知識源 (Knowledge Source : KS) にモジュール化することができる。これにより、知識の整理が行いやすくなるし、実行時に同時に処理すべきルール数が減少し処理スピードの効率化が図れる。ルールからの作業領域中のフレーム要素へのアクセス機能としては生成、削除、変更、パターン指定による探索等がある。また、関数定義領域内のメソッドを含むESPプログラムがルール中に記述可能である。知識源単位ではルール適用の区別 (前向き/後向き/混合)、コンフリクト解消の戦略、知識源のEXIT条件、証明すべきゴール (後向きの場合) 等の推論制御が指定できる。

知識源の選択に関する知識を我々はメタ知識と呼んでいる。メタ知識は前向きのルール形式 (メタ・ルール) で記述する。メタ・ルールの左辺はWHへのパターン・マッチングの指定を行う。右辺ではWH変更の指定とともに実行すべき知識源を、さらに、その知識源が後向きの場合は証明すべきゴールを指定する。メタ・ルールによって、メタ知識を非決定的に記述することができるし、ルールの書き方によっては決定的に (手続き的に) 記述することも可能である。

ESPによるインプリメントでは知識源を個々のESPクラスへ変換し実行する形式を取る。

4. おわりに

PSI 上のエキスパート・システム開発支援のプロトタイプ・ツールの開発について述べた。開発は現在、実装段階にあり、今年度中に製作の完了と実際にアプリケーションをのせての評価を行う予定である。今後は、知識表現方式、推論方式、知識獲得支援機能、知識ベース管理、知的インタフェースなどにおいてFGCSプロジェクトの成果技術の取り込みをおこないながら機能拡充をはかっていく予定である。

[参考文献]

- [1] Buchanan, B.G., "Research on Expert Systems", HPP-81-1, Department of Computer Science, Stanford, 1981
- [2] Buchanan, B.G., and Shortliffe, E.H., "Rule-based Expert System: The MYCIN Experiments of the Heuristic Programming Project", Reading, Massachusetts: Addison-Wesley, 1983.
- [3] Hayes-Roth, F., Waterman, D.A., and Lenat, D.B., "Building Expert Systems", Addison-Wesley, 1983.
- [4] 久保野等, 「PSI 上のエキスパートシステム開発支援ツール (2)」, 第33回情報全国大会5K-2, 1986, 10
- [5] 永井等, 「PSI 上のエキスパートシステム開発支援ツール (3)」, 第33回情報全国大会5K-3, 1986, 10
- [6] 進藤等, 「PSI 上のエキスパートシステム開発支援ツール (4)」, 第33回情報全国大会5K-4, 1986, 10

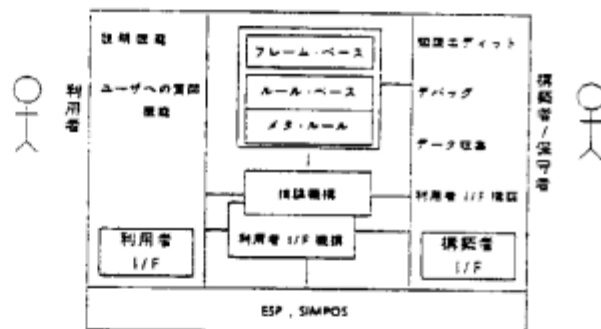


図1. ツールの全体構成

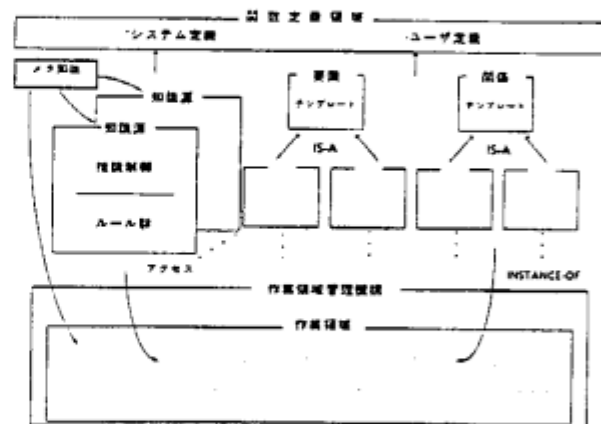


図2. ルール・ベースとフレーム・ベースの戦略構成

久保野 秀雄、沢本 潤、永井 保夫、岩下 安男 (財) 新世代コンピュータ技術開発機構
進藤 静一 (株) 三菱電機

1. はじめに

エキスパート・システムに於て、問題解決の際に扱われる問題領域、対象の表現手法として、対象の構造の見通しの良いフレーム型の表現言語が有る。PROTONの対象表現環境、Fact/Model representing Environment (以下、FMEと呼ぶ)は、フレーム型の表現形式を基本とし、構成要素の状態と構成要素間の関係に依って、対象を表現・操作する環境である。以下に、FMEの構造、対象の表現及び機能について説明する。

2. FMEの構造

ユーザーの設定する対象は、予め設定されたテンプレートを用いて生成された実体群として、Working Memory (以下、WMと呼ぶ)上に表現される。対象の設定、変更等は、WMの管理機構であるWorking Memory Management System (以下、WMMSと呼ぶ)を介して行われる。

対象の表現に先立って、構成要素及び関係の内部的定義となるテンプレートを設定しなければならない。要素テンプレート (以下、TEと呼ぶ)、関係テンプレート (以下、TRと呼ぶ)は共にフレーム型の表現形態を採っている。各テンプレートの具体化に依って生成される各実体は、そのテンプレートに依って管理される。(図1参照)

3. 要素表現

TEは凡そ、テンプレート名とその親テンプレート名(複数可)、及び一群の属性-属性値から構成される。各属性には、デフォルト値として、数値、文字列の他に、その値の検索、更新の際に起動されるデモンの記述が出来る。以下に、TE表現の特徴を示す。

継承、包含関係

継承機構は、対象の表現に当たって採用しているテンプレートの階層性に於いたメカニズムであり、要素実体及び関係実体の何れにも関係してくる。エキスパート・システム構築上、多用される属性継承、実体リンクについては、FMEで、super、has-partという形で予め用意している。TEの属性継承では、super定義にあるテンプレート間で、属性に関する情報の継承が、ファセット単位に行われる。尚、TRでの継承は、制約条件、意味情報、置換式情報等に行われる。子で当該情報の明示が為されている場合には、その情報が上書きされる。親は複数存在しても良いが、その定義順に順型探索が行われる。親にその属性が存在した場合、探索は打ち切られる。

包含関係は、has-part定義に依る実体リンクに依って実現する。has-part定義中に、実体生成情報が設定されている時には、当該TEより生成される実体に、生成情報に従った実体が付加される。has-part定義の実体が消去された際には、アクセスはエラーとなる。共に定義情報の参照が可能である。即ち、superに関しては、任意の概念に於て、その上位概念の上位概念は、その上位概念であり、has-partに関しては、任意の実体に於て、その部分実体の部分実体は、その部分実体である。

付加手続き

対象には、要素テンプレート・レベルで属性ごとに、固有の付加手続きが記述出来る。付加手続きは、当該属性へのアクセス、もしくはそれを含む実体の生成が為された時

に起動され、一通の手続き若しくはユーザーからの応答に依って、Valueの検証若しくは設定・修正を行う。この手続きは、以下の3つの場合に大別される。

- 属性値が、何らかの制約条件を持つ。
- 属性が、他の属性に束縛されている。
- 属性値が、予め設定出来ない。

Valueの制約条件チェックは、属性値の変更時に実行される。チェック結果がエラーの際、付加手続きで例外処理が指定されていれば、その手続きを行う。

付加手続きの処理の優先度はデフォルト値の設定よりも高く、継承関係が有る場合には、自分をも含めて、最も身近な手続きから行い、成功した段階で残りの手続きは、放棄される。付加手続き中での当該実体の属性値指定は、当該属性名を直接埋め込むだけで良い。Multiple Valueの場合は、括弧付きで位置指定を行う。その実体に包含されている実体の属性値指定も可能である。包含されている下位実体に、更に下位の属性参照用の便宜的な属性を挟む事に依り、包含関係にある任意段下位の属性の情報を、付加手続きを用いて活用出来る。また、予約語 u-set を設定した場合には、当該実体の全属性とその値がウィンドウに提示されると同時に、当該属性の値の更新がユーザーに要求される。(図2参照)

4. 要素間の関係表現

TRは凡そ、テンプレート名と関係の意味情報及び一群の置換式から構成される。各TRより生成される関係実体は、TEより生成される要素実体間の関係を表現する。以下に、TR表現の特徴を示す。

任意項関係の定義

対象内要素実体の関係は、関係定義のテンプレートからの実体の生成、及びその実体の修正、削除に依って定義される。関係実体の適用範囲は2項関係に留まらず、任意項の関係を表現し得る。FMEでは、任意項関係の記述の冗長性を排除する為に、関係を1つの実体として、引数となる要素実体の外に設定している。関係実体生成時に、テンプレートで規定される引数に、要素実体名が設定され、ルールのマッチング・パターンでは、その引数リスト内の引数の位置で、関係内の要素実体を同定し得る。

関係の意味の定義

ここで定義された要素間の関係実体は、要素実体間に単にある関係名称を割り振るのではない。その関係の意味付けを定義出来る。意味情報は、当該関係実体に依ってリンクされた要素実体間の、属性レベルでの対応関係定義用デーモンである。記述は、述語形式を取る。引数に対応する要素実体の当該属性がアクセスされた時に起動され、残りの要素実体の当該属性に、制約条件チェックが掛けられる以前に、副作用を引き起こす。推論過程で引数となっている要素実体に属性レベルで変更が行われても、関係の意味に従って、引数となっている要素実体間での関係を保証する。

等価関係の組合わせ情報の定義

関係実体の記述レベルは、Prologでの変数のないヘッド程度に過ぎないが、テンプレートに記述された置換式に

Expert System Shell on PSI (2) ~ Fact/Model representing Environment ~
H. Kubono, J. Sawamoto, Y. Nagai, Y. Iwashita, S. Shindou (*)
ICOT, (*) Mitsubishi Electric Corporation

依って、検索要求の出された関係が、単一の实体として存在しない場合でも、実体の組み合わせに依って、要求された関係の検索が可能となっている。この組み合わせでは、AND-OR結合を実現している。(図3.参照)

5. 機能

WM上に設定される対象を操作するに当たって、幾つかの有効な機能をFMEは用意している。以下に、その機能について説明する。

実体の生成、変更、削除

対象は、要素及び関係定義のテンプレートの具体化に依って実現され、WM上に格納される。生成された各実体には、WM内での生成時点を示すタイム・タグと呼ばれる実体識別子が自動的に付与される。タイム・タグは、実体の更新の際にも、その時点でのWM内で最新のものに更新される。この識別子は、発生可能なルール群の競合解消時に使用される。要素実体の生成に際しては、各属性にテンプレートに記述されたデフォルト値が初期設定される。デフォルト値の設定が無い場合には、属性値には、ストリング"null"が設定され、当該属性値を参照した場合、マッチング用の変数は、束縛されない。推論過程での対象の変更結果は、要素実体の属性に関する情報、及び要素間に設定されている関係実体の追加、変更、削除に依って表現される。

実体の探索

ルールからの成るパターンを持つ実体の探索要求に対して、FMEはタイム・タグ情報を参照して、WM内の最新の实体から探索していく。その際に、要求実体のテンプレートが既知であれば、テンプレートを指定する事に依り、探索範囲の絞り込みを行う。これに依り、探索処理の高速化が図れる。当該テンプレート上に存在しない場合には、その下位のテンプレート群から該当する実体を探索する。その為に、FMEはWM内のファクト・テンプレートの階層情報を管理している。

関係実体の探索に關しては、更に置換式に絡んだ処理メカニズムを用意している。任意の要素実体間(要素実体の個数>0)に関する任意の関係の探索要求に対して、該当する関係実体が存在しない場合、当該関係テンプレートは、予め設定された置換式に基づいて、FMEへ探索要求の出された関係と等価な関係の組み合わせの探索を要求する。組み合わせられた関係に關しても実体が存在しない場合には、更にその関係と等価な関係の組み合わせの探索が行われる。

このように、FMEは、要求された関係を検証するまで関係の組み合わせを展開する。その際問題となるのは、探索のループである。関係の組み合わせが展開されていく過程で、既に該当する実体の無い事が明らかにされた関係への探索要求が発生した場合、そのままでは探索経路が無限に延びてしまう。FMEは、それを回避する為に、探索経路を常に監視し、既に該当する実体の無い事が明らかにされた関係への探索要求が発生した場合、その要求をFAILさせバックトラックを行う。この関係の探索処理の際には、その関係と等価な如何なる関係の組み合わせに依っても、その関係が存在しない時に、下位の関係を探索する。

6. 今後の課題

実体内部での手続きの他に、実体間での情報の様々な処理形態が考えられる。FMEでは、階層的モデリング、通信環境の実現を検討している。

【参考文献】

- [1] 沢本等、「PSI上のエキスパート・システム開発支援ツール (1)」, 第33回情報処理全国大会 5K-1
- [2] 永井等、「PSI上のエキスパート・システム開発支援ツール (2)」, 第33回情報処理全国大会 5K-2
- [3] 永井等、「PSI上のエキスパート・システム開発支援ツール (3)」, 第33回情報処理全国大会 5K-3

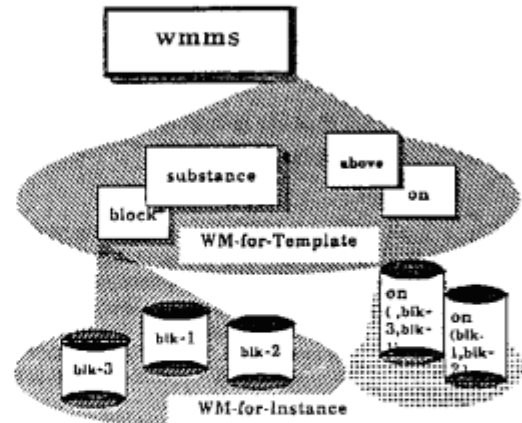


図 1. FMEの構成概略

```
te steel-block-with-bar
super
block
attrib
material default "steel" &
grav default 5 &
weight
if-gotten, arbit, before
weight
<- grav * (size,1) * (size,2) * (size,3)
has-part
bar { bar: (material, "steel") }
end.
```

図 2. TE外部表現例

```
tr on
super
above, contact:
restrict2(1, substance)(2, substance):
(1, position-z) >= (2, position-z)
interpret (1, position-z, new)
<- (2, position-z, new)
+ ((1, position-z, old) - (2, position-z, old))
interpret (2, position-z, new)
<- (1, position-z, new)
+ ((2, position-z, old) - (1, position-z, old))
on ([upper, lower])
subst
above ([upper, lower]) &
contact ([upper, lower])
or
on ([upper, lower])
subst
under ([lower, upper])
end.
```

図 3. TR外部表現例

PSI 上のエキスパートシステム 開発支援ツール (3)

5K-3

— 推論処理部 —

永井 保夫、澤本 潤、久保野 秀雄、岩下 安男

(財) 新世代コンピュータ技術開発機構

渡邊 勝一

(株) 三菱電機

1. はじめに

本論文では、PSI 上のエキスパートシステム開発支援ツールPROTON[1]の推論処理(ルールベース)部について述べる。推論処理部の特徴は知識源(Knowledge Source:KS)とKSの制御を行なうメタ知識という概念を導入することで、推論処理のモジュール化と推論方式の柔軟な切り換え機能を備え、かつKEに対しては容易な知識ベースの構築可能な環境を提供することを目的としている。以下では、メタ知識と知識源について説明する。

2. メタ知識

メタ知識とは知識を利用するための知識、つまり推論を制御する知識でありルールベース・システム、フレームベース・システム、両者を統合したシステムなど様々なアーキテクチャを有したシステムに取り入れられている。多数のルールベース・システムやフレームベース・システムではメタ知識がメタルールやメタフレームといったメタ用の階層的な表現形式となっており、メタ知識と処理対象に関する知識の分離のなされる場合が少なくない。本システムでも同様に適用対象となる問題をサブ問題に分割して解決する(複数の知識源に分割して問題解決を行なう)場合、それぞれの処理(知識源)を制御する知識をメタ知識として分離し、システム全体の見通しを良好なものとしている。

表現形式

メタ知識は、メタルール、メタルールの制御戦略(first hit strategy 又は、CRS)、メタルールの適用-実行サイクルの停止条件などから構成されており、メタルールの形式は現状ではプロダクションルールの前向き推論用ルールと同様にしている。メタルールの付加情報にはルールの適用についての指定及びルールの適用優先度がある。メタルールの条件部はWorking Memory(WM)登録パターン、ユーザ定義の述語(メソッド)又はESP(KLO)述語などのand結合となっており、実行部はWM登録取り扱い用述語、ユーザ定義の述語(メソッド)又はESP(KLO)述語などのand結合と知識源名(但し、KSが後向き推論を行なう時は後向き推論に対して与えられるゴールパターンも付加される)から構成される。

知識源(KS)の制御

知識源(KS)の制御はメタ知識とメタ知識用の推論機構を用いて行なうものである。メタ知識をメタルールという記述形式にしたのは、メタルール表現により様々なKSの起動-実行メカニズムが実現可能だからである。この場合、ユーザはルールによる推論機構で不可欠となるWMの設計をKSの制御用とドメイン用に使い分けて設計することが好ましい。

KSの制御(つまり、メタ知識の推論)のメカニズムを要約すると以下ようになる。

1. メタ知識(ルール)から適用可能なものを1つ選択
2. 選択されたメタ知識(ルール)の実行部に記述されているKSを実行
3. KSの実行結果がsuccessの場合(FC用またはMIXED用は常にその結果がsuccess、BC用の結果はsuccess/fail)、メタ知識(ルール)の実行部に記述されたアクション(通常、WM上の制御情報に副作用を発生させる)を実行させる。結果がfailの場合には何もしない。

4. 1へ戻る

さらに、ユーザがこのようなメタ知識を拡張して使用すればアラックボードモデルの実現も可能である。(但し、その場合WM機能も拡張して用いる必要がある)。

上記の項目1と2はそれぞれ先程述べたメタルールの評価(選択)-実行(recognize & act)サイクル中の評価(選択)と実行に相当する。これには、アジェンダ(conflict set)を生成しないでメタルールの記述順序だけに依存した形でルールの条件部が満足されたものを、即ち、選択する場合と、すべてのルールの条件部を評価してアジェンダ(conflict set)を生成し、何らかの評価に基づいてその中から1つを選択する場合の2つが与えられる。本システムでの後者のケースにあたるCRS(conflict resolution strategy)は、OPSSのLEX戦略とはほぼ同様でありここでの説明は省略する。

最後に、KSの制御メカニズムについての概略を図により説明する。メタ知識中のmeta_rule1が選択されて、それを実行すると知識源1が起動され結果として知識源中のルールが実行される。知識源がsuccessして終了した場合にはmeta_rule1の右辺のstantkが実行され、WM中のKS制御情報にupdate(delete)される。これがKS制御の基本サイクルとなっている。ユーザはオプションとして、メタルールの選択-実行サイクル毎に停止条件をチェックできるような記述が可能である。

3. 知識源

知識源を導入したメリットはルールを複数の知識源という形式にグループ(モジュール)化して問題解決を行なう事で探索空間の絞り込みによる推論処理の効率化を図ることである。本システムで述べているKSは各々が独立して問題の解決を行ない、1つのKSの実行中に他のKSをよびだすという事は行なわない。又、対象となる問題が複数の問題に分割できない(複数のKSに分割不可能な)場合はKSを1つとみなして処理をおこなう事も可能である。推論方式はKS毎に決定されており、後述するFC、BC、MIXEDの3種類が許されている。

表現形式

KSの推論制御情報はルールの適用-実行サイクルの制御戦略(CRS 又は、first hit strategy)、推論形式(FC、BC、又はMIXED)、及びルールの適用-実行サイクルの停止条件から構成されている。さらに、KSの推論形式がBCの場合には、そのKSに静的にゴールを指定することも可能である。

ルールは処理対象に対する手続的な表現であり、関連したものがKSとしてまとめられ取り扱われる。ルールの形式にはFC用とBC用の2種類があり、先程述べたKSの推論方式がFCの場合KS中のルール表現はFC用ルールのみで記述され、BCの場合にはBC用ルールのみから記述され、MIXEDの場合KS中のルール記述はFC用ルールとBC用ルールが混在して記述される。ルール記述にはルールの条件部と実行部以外に付加情報として優先度やルールの適用記述の指定が記述可能となっている。

KSが前向き推論を行う、つまりKS中のルールがFCだけから記述されている場合、実行部はユーザ定義の述語、ESP(KLO)述語やWMのupdate処理を行なう組込み述語make、modify、removeという3つの述語などから構成されている。

BC用ルールの構成もFC用ルールとほとんど同じである。但し、注意しなければならないのは、FCと異なってルールの実行部がWM登録のパターンであるか又はpropose 述語(引数としてWM登録のパターンが含まれている)である点である。前者の場合はゴールに対して後向き推論が行なわれてもWMへアサーションされない(副作用がない)。しかし、後者のpropose 述語をルール中で利用した場合はBC

Expert System Shell on PSI (3) -- Rule-Base System Environment --

Y. NAGAI, J. SAWAHOTO, H. KUBONO, Y. INASHITA, S. SHINDOU (*)

ICOT, (*)Mitsubishi Electric Corporation

推論の結果としてWMに副作用が起こりWMへのアセーションが行なわれる。ユーザはこのような機能を選択したルールの記述が可能である。また、BCルールの実行中に、ask user (query) 機構を起動してユーザに対するファクトに関する問い合わせを行なうか行わないかはファクト中のテンプレートに付加手続きとして指定する。

WM要素パターンは対象表現部における構成要素を3つ組形式として表現するか又は構成要素間の関係を述語として表現する。前者はエレメントと言われるシステムの構成要素を属性-属性値で表現しており、1つのエレメントに対して複数の属性-属性値群が記述可能である。対象表現部における構成要素表現ではTE間にhas_a 関係が定義可能となっている。TE間での要素(実体)継承は、上記の関係に基づいて行なわれる。このような要素継承関係をルール・シンタックスでは一般にWM要素パターン中の属性名で表現できる。後者の構成要素間の関係は、二項関係や多項関係が記述可能である。このようなエレメントやリレーションはテンプレートとそれによって生成される実体からなっている[2]。

推論形式

KSで実行される推論形式には次の3種類がある。

- ・forward chaining (FC)
- ・backward chaining (BC)
- ・mixed strategy (MIXED)

各処理手順について以下に示す。

前向き推論(FC)はデータ駆動型推論又はパターン主導型推論と呼ばれ、データが入力されることによりそれに適したルールの選択-実行が行なわれ、その結果WM中のWM要素が更新されるという処理の繰り返しである。代表的な例としてはOPS5などがある。

後向き推論(BC)は目標駆動型推論と呼ばれ、目標を設定しこれに従って後向きルールを起動したりその際にユーザの問い合わせを行なって目標の決定(検証)をする。代表的な例としてはMYCINがある。

混在型推論(MIXED)は前向き推論で仮説を生成して、各仮説に対して後向き推論を適用して仮説の絞り込みを行なっていくという処理の繰り返しである。本システムの混在型推論はFC主体で動作するようにしており、FCの実行に際して必要な情報を得るためにシステムが自動的に(補助的に)BCを起動するものである。

ルールの適用方式

KS中のルールの適用方式にはfirst hit strategyとCRSの2つがある。従って、3種類の推論形式に対して上記の適用方式が考えられる。しかし、FC及びBCのルールの適用メカニズムは省略して、混在型についてのみ説明する。

混在型はFCの実行に際して、必要な情報を得るためにシステムが自動(補助)的にBCを起動するというメカニズムをとっており、BC起動タイミングと混在型推論自身の制御は現在検討中である。しかしながら、以下には具体的実現の一例として、first hit strategyとCRSについて述べておく。

1. first hit strategy

- 1) FC用ルールだけを評価(適用)-実行する。
- 2) 適用可能なFC用ルールがなくなったら、すべてのFC用ルールの条件部のパターンからゴールパターンを生成し、BC用ルールを使ってこれらのゴールパターンを検証する。
- 3) WMに副作用が起こった場合には、1)に戻り、BC用ルールだけの選択-実行サイクルを繰り返す。そのサイクルで停止条件が満足された場合、混在型推論は終了する。
- 4) BC用ルール実行後、WMに副作用が起こらなったり、FC用ルール実行中に停止条件が満足された場合に、混在型推論は終了する。

2. CRS

- 1) FC用ルールで条件部が発火可能なものをすべてconflict set (agenda)に収容し、その中の1つを選択-実行するというサイクルを繰り返す。

- 2) 次に、conflict set (agenda)中のFC用ルールが空になったら、すべてのFC用ルールの条件部のパターンをゴールパターンとしてBC用ルールを起動する(WM要素パターンの否定の場合はWM要素パターンをゴールパターンとする)。

- 3) BC用ルール実行の結果として、WM中に副作用が発生した場合は、1)に戻り、FC用ルールだけの選択-実行サイクルを繰り返す。そのサイクルにおいて、条件が満足された場合、混在型推論は終了される。

- 4) BC用ルールの実行結果としてWM中に副作用が発生しなければ、混在型推論は終了される。

4 おわりに

メタ知識を利用して柔軟な知識源の制御を行う推論制御(ルールベース)部について説明した。現在、推論処理部は実装段階に入っており、今後はブラックボードモデルの実現などアプリケーションを通じて種々の問題に適したメタ知識の記述とその適用-実行方式の検討を行っていく予定である。

参考文献

- [1] 澤本等、「PSI上のエキスパートシステム開発支援ツール(1)」、第33回情報全国大会5K-1、1986、10
- [2] 久保野等、「PSI上のエキスパートシステム開発支援ツール(2)」、第33回情報全国大会5K-2、1986、10

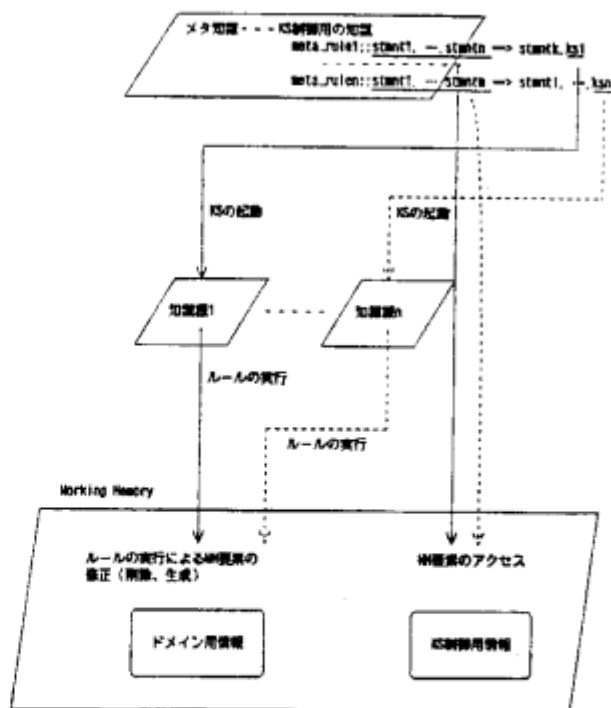


図 3. 混在型推論メカニズムの概略

PSI 上のエキスパートシステム 開発支援ツール (4)

5K-4

～ユーザーインタフェース部～

澤本 潤、久保野 秀夫、永井 保夫、岩下 安男

(財) 新世代コンピュータ技術開発機構

(株) 三菱電機

遠藤 静一

1 はじめに

本稿では、PSI 上のエキスパートシステム開発支援ツール PROTON [1] のユーザーインタフェース部の機能について述べる。ユーザーインタフェース機能とは、「ユーザーがシステムを利用することを支援する諸々の機能」である。エキスパートシステム開発支援ツールの場合、ユーザーとして、次の2種類を考える。

Knowledge Engineer (以下KEと略記)

ツールを用いてある特定の分野に関するエキスパートシステムを作成する者である。

End User (以下EUと略記)

KEによって作成されたエキスパートシステムを利用する者である。

また、ツールの利用段階として次の5種類を考える。

(1) 知識表現時

(2) システム起動時

KEやEUが、上の段階で作成された知識データをツールにとりこみ、推論を開始するまでの段階である。

(3) デバッグ時

(4) 通常実行時

EUが、KEによって作成されたエキスパートシステムを動かすことにより利用する段階である。

(5) システム評価時

ツール作成者、又は、KEが、システム実行時中の諸々のデータを収集することにより、ツールそのものの処理性能、又は、ツール上の知識ベースの処理性能を評価する段階である。

以下(1)、(3)、(5)について説明する。

2 知識表現時に必要な機能

エディタ機能とブラウジング機能を考える。エディタは知識をルールやファクトとして表現する為に必須である。ブラウジングではKEに今迄表現されたルールやファクトの大域的な情報を提示する。表示例を以下に挙げる。

・TE間、又はTE間のis-a関係による階層のDAG(Direct

ed Acyclic Graph) 表示

・あるTEに継承されるスロット及びその初期値及びデモンの表示

・ルールが参照しているTE、TRの表示

デバッグの効率向上の為、ここで挙げた諸機能の提供する情報はデバッグ時にもアクセス可能とする。

3 デバッグ時に必要な機能

デバッグとは、バグの発見と修正のくり返しである。この行為を支援する為にデバッグのもつべき機能は、システムを「とめる」、「みる」、「かえる」の3種類にまとめられる。以下、各々について説明する。

柔軟な動作制御「とめる」

可能な限り、システムの動作中の任意の時点で、その動作を中断し、再開させる機能である。中断時に、KEはシステムの内部状態を「みる」こと、および「かえる」ことが可能である。具体的にはステップバイステップ動作スバイポイントの設定である。スバイポイントは、ルール、WH、ゴールに設定可能である。

システムの内部状態のprobing = 「みる」

システムの中断時、又はシステムの動作中継続的に、システムの内部状態、又はその変化をKEにわかり易く表示する機能である。

トレース

1ルールの実行毎に、内部状態の変化を表示する。表示内容は、アジェンダ、発火ルール名、ルールの実行に伴うWHの変化、である。

WH

WHの構成要素はTEのインスタンス及びそれらのTEのインスタンス間で成立する関係である。この関係はTRを述語としたn項関係である。KEはWHに対して色々な見方をもつ。例えば、あるTEのインスタンスのみに注目したい場合、あるTRに対応する関係に注目したい場合等である。WHのprobingでは、KEの色々なWHへの見方を対処する為に、図1に示す探索木を用意する。KEは

Expert System Shell on PSI (4) -- User Interface --

J. SAWAMOTO, H. KUBONO, Y. NAGAI, Y. IWASHITA, S. SHINDO(*)

ICOT, (*)Mitsubishi Electric Corporation

この木を自由にトラバースできる。下位レベルのノード程、局所的な情報を与える。

推論の履歴

現時点までに行なわれた推論の履歴を表示する。表示レベルとしてKSとルールを用意する。前者では今迄に実行されてきたKS名を実行順に表示する。後者では、前向きKSに於ては、実行されてきたルール名を実行順に表示し、後向きKSに於ては、証明過程をAND-treeで表示する。

内部状態の修正「かえる」

システムを「とめる」、又は、「みる」ことにより発見されたバグを一時的に修正すること、又は、バグを発見する為に実験的にシステムの内部状態を一時的に変化させることを支援する機能である。修正対象は、WM、ルールである。

WMに関して

WM要素の追加、削除

WM要素のスロット値の書き換え

WMの過去のある状態への復帰

WMを過去のある状態に戻す。現在実行中のKSで過去に実行されたルールの実行直前の状態、又は、過去に実行されたKSの実行直前の状態に戻る。

ルールに関して

ルールの追加、削除

推論動作の強制終了

次に実行されるルールの選択、等。

4 システム評価時に必要な機能

以下のようなデータを収集することにより、システム評価に用いる。

ルールの実行スピード

1秒間のルールの発火回数である。測定の単位はユーザーの指定したKSである。このデータは推論方式のちがいが知識ベースの大きさによるルールの実行スピードの目安を得ることができる。また、他のツールとの性能比較のパラメータとなる。

WMの変化の履歴

アジェンダの履歴、および、ルールの実行によるWMの変化の履歴である。このデータにより、WMの参照の局所性（例えば、「あるルールでWMに追加されたWM要素が、次に発火するルールで参照される」といった規則性）が見い出せれば、WMのアーキテクチャに反映できると考えられる。

ルールへのアクセス頻度

KSの各ルールに関して、前向きKSの場合は、ルールのアジェンダ登録回数、ルールの発火回数を、後向きKSの場合は、ルールの参照回数、ルールのsuccess回

数を収集する。このデータにより、参照回数、または実行回数が0のルールや、逆に、頻繁に実行されるルールを知ることができる。

5 おわりに

PROTONは現在、実装段階にある。実装に関しては、PSIの提供する諸々のutility、具体的には、マルチプロセス環境、マルチウィンドウ、メニュー、ホワイトボード等の機能を利用し、快適なユーザー・インタフェースをめざす。

参考文献

- [1] 沢本等、「PSI上のエキスパートシステム開発支援ツール(1)」、第33回情報全国大会5K-1、1988.10



図1 WMのprobingに用いられるmenu network