4B-7

エキスパートシェル MIND-1 ―― 知識表現とユーザインタフェース ――

山本 雅基　谷口 久衛　藤井 敬久　丹羽 雅司
日本電装株式会社

1　はじめに

エキスパートシェル MIND-1 は、ファジイ推論を含む豊富な推論機能を有し C 言語により高速に動作するエキスパートシステムを開発するために作られた [1]。

MIND-1 ではルールを日本語で記述するので、DE(*Domain Expert*) は KE(*Knowledge Engineer*) の手を借りずにある程度自力で知識ベース開発できるようになっている。本稿では、MIND-1 の知識表現方式とユーザインタフェースについて報告する。

2　知識表現方式

MIND-1 では知識を日本語表記したプロダクションルールとして表す。ルールはルール名・プライオリティおよびルール節 (複数の前件節・後件節) からなる。これにより前向き推論・後向き推論およびファジイ推論用の知識を表現する。図 1にルールの例を示す。ここでルール節は「語句」と呼ぶ日本語句をならべて記述されるので、日本語表記をしている。したがって、ソフトウェアの知識のない DE でも独力でルールの作成・保守をすることができる。

ルール名：在庫ルール 2
プライオリティー：10
もし
品番 A の完成品の在庫量が 100 以下である。
品番 A の仕掛りが可能である。
ならば
品番 A を生産する。

図 1: ルールの例

2.1　語句とルール

MIND-1 では最初に対象領域の知識整理を行い知識の表現に使用する言葉を抽出する。たとえば、製品の在庫量を基準として生産を制御する問題では、「品番 A」「の完成品」「の在庫量」「を生産する。」などという言葉が抽出される。これらの言葉を「語句」と呼ぶ。あらかじめ知識整理を十分に行うと、ルール記述に使用する必要十分な「語句」を揃えることができる。また、日本語の語順は柔軟である。したがって、「語句」を順に右から修飾させるように並べることで必要なルールをもれなく記述することができる (図 2参照)。

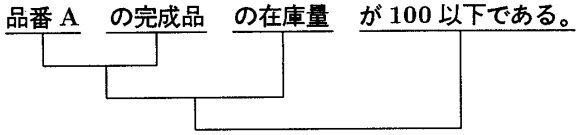

図 2: ルール節の例

これから、「語句」は図 3のように分類される。

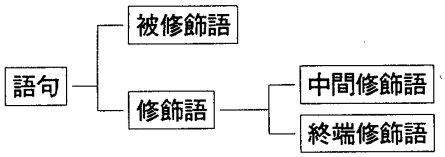

図 3: 語句の分類

被修飾語はルール節の左端に現れる語句であり、一般に名詞である。**中間修飾語**はルール節の途中に現れる修飾語であり、左側の名詞 (句) を修飾し新たな名詞句を生成する。**終端修飾語**はルール節の末尾に現れる修飾語であり、左側の名詞 (句) を修飾し前件節では真偽の判断やファジイ値の演算をし、後件節では動作の実行をする。ただし、修飾語は何にでも修飾できるわけではない (たとえば、「品番 A」に「が可能である。」を修飾させることは無意味である)。そこで、修飾語は修飾可能な名詞 (句) に関する情報をもっており、同じ名詞 (句) を修飾するものが同じグループを作っている。

このようにして作られた語句は次のようにオブジェクト指向概念で整理することができる。

(1) 被修飾語および名詞句：インスタンス
(2) 修飾語：メッセージ
(3) 修飾語を修飾させる：メッセージセンディング
(4) 修飾結果：メッセージの戻り値
(5) グループ化：インスタンスとメッセージをクラス概念でまとめる

そこで、MIND-1 ではこれらの語句を Obj-C 言語でプログラムする。Obj-C 言語とは C 言語をオブジェクト

Expert Shell MIND-1 : Knowledge Representation And User Interface

Masaki YAMAMOTO, Hisamori TANIGUCHI, Norihisa FUJII and Masashi NIWA
NIPPONDENSO Co.,Ltd.

指向に拡張した自社開発の言語である。Obj-C 言語で書かれたプログラムはプリプロセッサにより C 言語のプログラムに変換される。

Obj-C 言語ではインスタンスのメモリ管理をユーザが行うこととして、ガーベージコレクションによる処理速度の低下を防止している。

つまり、MIND-1 では被修飾語に対するインスタンスは静的な記憶領域管理をする。これをスタティックインスタンスと呼ぶ。また、中間修飾語の修飾結果としてのメッセージの戻り値は動的な記憶領域管理をする。これをダイナミックインスタンスと呼ぶ。ダイナミックインスタンスのメモリは終端修飾語を作用させた直後に全て解放される。なお、メモリを解放する処理は次に説明する知識変換処理時に自動挿入される。

2.2 知識の変換

ルールを構成するルール節は一つづつ C 言語プログラムの関数に変換される。推論エンジンはこのルール節に相当する関数をルールの構成にしたがって順次呼び出す。関数への変換は次のように行われる。

Obj-C 言語で定義された「語句」は、プリプロセッサで C 言語に変換される。一方、ルール節は Obj-C 言語のメッセージセンディングプログラムに変換され、ダイナミックインスタンスの解放プログラムが自動挿入され、同じくプリプロセッサで C 言語に変換される。そして、これらをリンクすることによりルール節毎の関数ができあがる。以上の処理は次に述べるエディタが自動生成する make ファイルを実行することで行われる。

こうしてできたルール節の関数はテーブル管理され、推論エンジンから起動される (図4参照)。

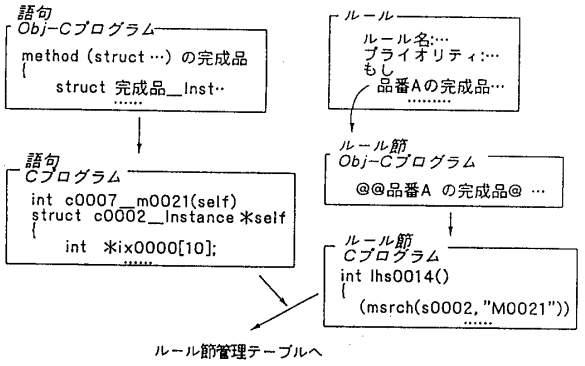

図 4: 知識の変換

3 ユーザインタフェース

以上のような知識表現方式により、知識ベースは「語句」と「日本語ルール」の2つに分離され、図5のように開発すればよくなった。つまり、KE は DE と話合いながら知識表現に必要な語句を Obj-C 言語で定義する。そして DE は自力で語句をならべて日本語文でルールを作成する。日本語文のルールは自動的に C 言語プログラムに変換される。このように KE と DE の役割分担が明確になり、知識ベース作成に DE が直接参加できるようになり開発効率が向上した。

そこで、KE および DE の作業を支援するためのユーザインタフェースとしてそれぞれ専用エディタを作成した。これらのエディタは X-Window を用いてマルチウインドウ構造となっており、使いやすさが考慮されている。

語句エディタは KE が語句を開発する時に使用し、クラス管理などオブジェクト指向言語である Obj-C 言語でのプログラム開発支援機能がある。

ルールエディタは DE がルールを開発する時に使用し、ルール節作成時に修飾関係を満足した語句を逐次自動的に表示するなど日本語でのルール作成を支援する機能がある。これにより DE はマウスで語句を選択して日本語のルールを作成するので、キーボードから日本語入力をする必要がなくなり作業効率が向上している。

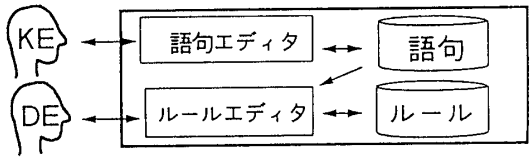

図 5: 知識ベース開発作業

4 まとめ

DE が日本語で楽にルール開発できる知識表現とユーザインタフェースを有するエキスパートシェル MIND-1 を開発した。

この知識表現により可読性の高い知識ベースを作ることができる。知識獲得手順が改良され、DE が直接知識開発作業に参加できるようになった。また、一度作成された語句は他の知識ベースを開発するときに再利用可能である。以上により、MIND-1 では知識ベース開発効率が向上している。

MIND-1 は C 言語で開発され YHP 社のワークステーションで動作する。現在 MIND-1 を使用して工程制御や生産管理を対象としたエキスパートシステムが検討中である。

参考文献

[1] 丹羽 他: エキスパートシェル MIND-1 — 概要 — 第 39 回情報処理学会全国大会論文集,1989