移植性を考慮したエキスパートシステム開発ツール

鬼頭繁治・塚田充広*

Portable Development Tool for Expert Systems

Shigeharu KITO and Mitsuhiro TSUKADA*

A portable development tool for expert systems specific to the domain of trouble diagnosis is developed based on the Pure Production System of Saito and Mizoguchi. The production system is extensively modified to have capability of global backtracking and rule base of recursive structure. The performance of the tool is partially verified by generating an small expert system for trouble diagnosis of motor cycles.

緒　言

1960年代に端を発する，コンピュータを利用して人間の思考過程をシミュレートしようとする人工知能の研究が，近年盛んになってきている。同時に，その人工知能研究の成果を実際の問題の解決に応用する，いわゆる知識工学の研究も進み，1980年代になって一段と注目されるようになっている。知識工学の代表的分野としては，ある領域の専門家に近い行動を目ざすシステムとしてのエキスパートシステムの開発があげられる。これまでにも限られたレベルではあるが実用的なものもいくつか作られてきている[1),2),3)]。

このようなエキスパートシステムの構築に際しては，システムごとに最適の実現方法を使用するのが普通であるが，このことは対象領域（問題）ごとに計画段階から独立なシステムを構築することを意味し，エキスパートシステムの応用の観点からは好ましくない。

一方，これまでも人的，時間的資源をかけずにエキスパートシステムを作製できるような開発ツールの作製も試みられており，対象分野は限られるものの，比較的容易にエキスパートシステムを生成しうるツールも部分的に商用化される段階となった[4)]。

しかし，これらの開発ツールはその使用に際してはハードウェアも含めたコンピュータシステム環境を特定している場合がほとんどであり，手近で使用するのは困難である。

ここでは，この欠点を除去する試みとして，工学的システムの故障診断エキスパートシステムの生成ツールを作製したので報告する。

* 現在，日本ディジタルイクイップメント㈱勤務

1．システムの設計

ここでは，以下の特徴を有するエキスパートシステム開発ツールの作製を目的とした。

(1) できるだけ広範囲な計算機環境のもとで使用可能であること。

(2) 上記の目標を満足するため，小型のコンピュータ上でも使用可能とするべく，できるだけコード量を小さくする。

(3) 生成するエキスパートシステムは故障診断用と限るがその範囲内では汎用性を持たせる。

一方，以上の機能を優先するため，以下の点については問題はあるものの，特に注意は払わなかった。

(1) ユーザー（開発ツールの使用者）インターフェース。

(2) 実行効率などのシステムパーフォーマンス。

このうち，(1)のユーザーインターフェースについては開発ツール研究のすう勢とは逆行することとなり大いに問題とはなるが，ここでの開発目標達成のため止むをえないものとした。

以上の性質を持つシステムであれば，知識工学の知識を有する者であれば，手近にかつ容易に故障診断エキスパートシステムを構築できるであろう。

以上の性格を有する開発ツールを作製するために，次のような構成を採用した。

1.1　推論システム

エキスパートシステムのような知識情報処理システムを実現する手段はいくつか考えられるが，生成されるエキスパートシステムごとに使用する知識が異なる点を考えて，ここでは知識ベースとそれを使用して推論する部分の分離が容易な“プロダクションシステム”（以後，簡

単に“PS”と略記する）を採用した[5),6)]。プロダクションシステム（PS）に沿って開発されたエキスパートシステム開発ツールとしてはOPS5[4)]が著名である。また，報告されているエキスパートシステムのほとんどがPSを採用している。

PSは知識を基本的には

IF　条件部　THEN　実行部

の形のルールで書き表し，その集合体としての“ルールベース”と，ある時点における“世界（状態）”を記述している“データベース”，およびデータベースに対してルールを適用してその内容を書き換える方法を制御する“インタープリタ”から構成される推論システムと言える。ここでルール中の“条件”がその時点でのデータベースの内容とマッチするとそのルールが起動され実行部がデータベースの内容を変更するというサイクルを繰り返し，データベースの内容が終了条件と適合して停止するものである。この場合中心となるのはパターン照合の実行である。

1.2　記述言語

システムの記述言語としてはLispを使用した。記述は，手続言語，Prologなどによってももちろん可能であるが，ここでは以下の理由からLispを使用することとした。まず，後述するように本ツールによってエキスパートシステムを構築するためには，対象分野のためのルール記述など，いくらかプログラムを変更することとなる。これらはほとんどリスト構造に係わることから，リスト処理言語としてのLispの使用が好ましい。つぎに，現在では多くの計算機環境下でLispの処理系が利用可能であり，特殊なLispの機能を使用しない限り移植性が良好である。さらに，同じくリスト処理に適したPrologの場合とは異なり，実用的なLisp処理系はコンパイラを備えたものが多く，実行効率の点から有利である。これらの点を踏まえてLispを使用することとした。

2．システムの実現

2.1　PSの構造

一口にPSとは言っても実際はエキスパートシステムごとに，対象とする分野に好都合となるようインタプリタ，ルールベースの表現法に工夫を凝らしているのが普通であり，汎用的なPSと言えるものは無いに等しい。

しかし適用分野を特定しない開発ツールにおいては，PSのインタプリタ部は簡単であることが望ましく，一方，ルールベースに関しては知識の構造が対象分野によって大きく異なることが予想されるので表現能力の高いことが必要である。したがってここでは，ルールを最初から一つづつ適用していく形式を持ち，かつ前向推論型である純粋プロダクションシステム（純粋PS）を基礎にした。この場合，インタプリタのフレームには斉藤・溝口[6)]により公開されているプログラムを使用した。

純粋PSは単純であるだけに多くの欠点も有する。なかでも対象分野に関する知識をルールベースとして表現する場合に制約が大きく，エキスパートシステムが対象とするような複雑な知識構造を有する場合には事実上ルール化が不可能である。したがってここでは，以下に述べるようにインタプリタ，ルールベースを改良した。この結果，基本的なフレームは斉藤・溝口[6)]のものと同じであるが，内容は非常に異なったものとなった。

2.1.1　インタプリタ

いかなる専門分野であれ，完全な知識を得ることは不可能である。したがって，エキスパートシステムの作製に際しては，相互の関係が不明確であり，相互に矛盾を含んだ知識によって構成されたルールベース上で推論を行い目的を達成することを念頭に置かなければならない。この点を考えると純粋PSの想定している場合とは著しく離れたものであることが明白である。ここでは，前述したように，実際の場合の複雑さの大きな部分はルールベースの表現能力の増大によって吸収する方法を取るわけであるが，インタプリタにも汎用性があると考えられるバックトラック機能を付加した。その結果，推論を進めていく上で，途中のルール選択に何らかの誤りがあり矛盾が生じた場合は推論を改めることができる。

ここで採用したバックトラック法は，選択されたルール中で実行可能でありながら現実には実行（採用）されなかった実行部を仮想的に実行させておき，その際書き換えられたデータベースを一時的にスタックにプッシュしておくこととし，矛盾が生じる度にそれをスタックからとり出すという方法である。このバックトラックはデータベース全体に作用することから，その汎用性は大である。このバックトラック法を模式的に図1に示す。図1において，R1，R2……は適用されたルール，D1，D2，D2′……はデータベース（′の表示のあるものが仮想的に実行された結果のデータベース），●印は矛盾の発生を表す。

2.1.2　ルールベースの構造

ルールベースはプロダクションルール（データベースの内容に対する書き換え規則）の集合であり，PSによる知識表現の要である。

複雑な構造を持った知識をルールベースに埋め込む方法は色々と報告されているが，本システムにおいてはルールの実行部にもルールを埋め込む再帰的方法を取ることとし，そのための関数群を提供している。すなわち，全体としては再帰的PSという形態を取っている。この

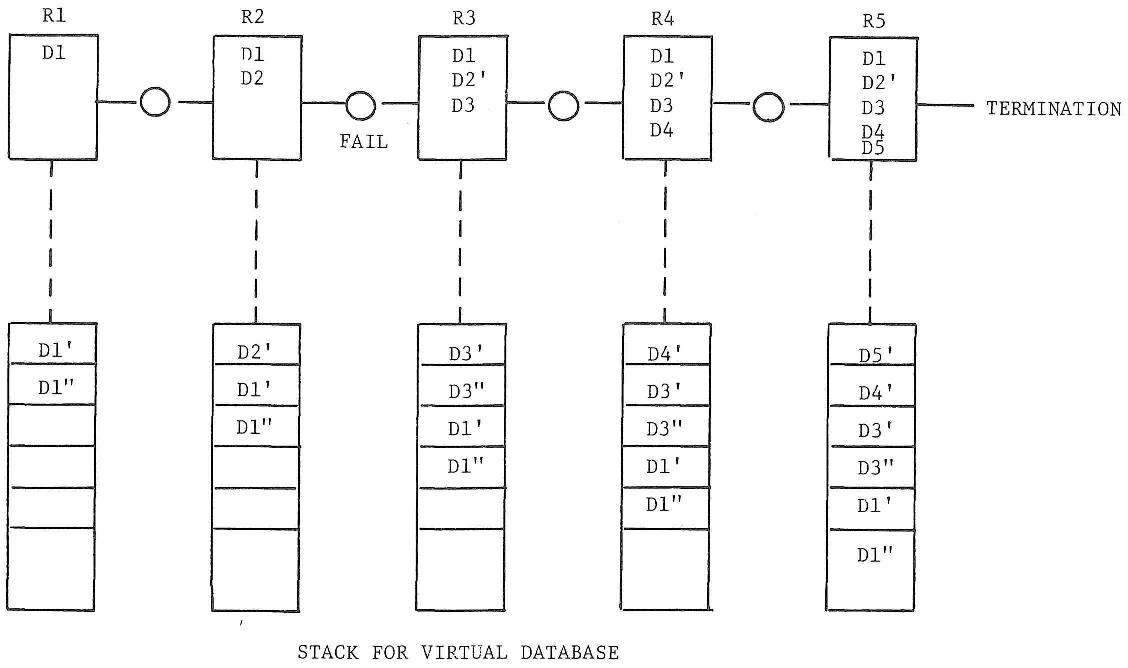

図１　ＰＳのデータベースによるバックトラック
（推論は左から右に進行する）

結果，実行部を実行する際にはある意味で仮説をたて検証しながら計算を進めるという柔軟なエキスパートシステムの生成も可能である。

一方，条件部は単なるリストとして表わされ，特別な構造は有していない。

2.2　プログラム

実現には最初はIBM370/138，CMS上のLisp F3[7]を使用し，インタプリタの大きさは清書した状態で約800行である。

3．使用例

本ツールの使用例としてはオートバイのエンジン，電気系統の故障診断用エキスパートシステムの構築を試みた。ルール数は診断対象が比較的簡単なシステムであることから49個と少数である。またルールについて特異的なことは，ほとんどのルールについて条件部のリスト要素が１個となったことである。図２に本ケースのルールの一例を示す。ルールベースに加え，エキスパートシステムのユーザーに対する状況質問関数を加えてオートバイ故障診断用システムとした。

このようにして生成されたオートバイ故障診断用エキスパートシステムの実行例を図３に示す。なお図３においてはユーザーへの質問の一部は省略してある。

```
(R190 ((LOWER COMPRESSION))
     ***
     ((MAKE-NULL ((LOWER COMPRESSION)))
      (LONG (10000)
            (KM)
            ((*MEMB-DB ((L COMP)) ((PLAN (CYLINDER NO MAMOU.)
                                          (CYLINDER ABRATION))))))
      (*MEMB-DB
         ((O-L MIX) (L COMP))
         ((PLAN (HEAD-GASKET NO FUKINUKE.)
                (HEAD-GASKET BLOW-BY))))
      (LONG (10000)
            (KM)
            ((*MEMB-DB
                ((L COMP))
                ((*MEMB-OR
                    ((KAITEN DOWN) (KAITEN UP))
                    ((PLAN (CLANK-SHAFT NO OIL-SEAL GA DAME.)
                           (CLANK-SEAL OR LEAD-VALVE)))))))))
```

図２　プロダクションルールの例

```
Q-20:(NEWTRAL DE KICK DEKIRU-KA ?)
>
N

Q-21:(PLUG GA WATER DE NURETE INAI-KA ?)
>
T

Q-22:(PLUG NO SAKI-KARA HIBANA WA DEMASU-KA ?)
>
N

Q-23:(ENGINE DE KIKINARENAI NOISE GA SHIMASI-KA ?)
>
T

/---PS-NO-DATA-BASE-WO-INPUT-SHITEKUDASAI-
>
(ES AT SOON)
*********************************************
DATA-BASE:(ES AT SOON)
          (NOISE)
          (C-L*CHANGE)
          (WATER-C)
          (CLUTCH-S)
          (S-K-UNIT NG)
          (PLUG CONTAMINATE)
          (PLUG-C (1 3))
          (KAITEN DOWN)
          (AIR-CLEANER CONTAMINATE)
          (PLUG UM)
          (O-Q NG)
          (C-AS S)
          (L COMP)
          (CHOKE)


*****MATCHED-RULE:R40*****
```

```
**********************************************
DATA-BASE:(PLUG CONTAMINATED)
          (** (ES AT SOON))
          (NOISE)
          (C-L*CHANGE)
          (WATER-C)
          (CLUTCH-S)
          (S-K-UNIT NG)
          (PLUG CONTAMINATE)
          (PLUG-C (1 3))
          (KAITEN DOWN)
          (AIR-CLEANER CONTAMINATE)
          (PLUG UM)
          (O-Q NG)
          (C-AS S)
          (L COMP)
          (CHOKE)

*****MATCHED-RULE:R200*****

**********************************************
DATA-BASE:(CARBON)
          (** (PLUG CONTAMINATED))
          (** (ES AT SOON))
          (NOISE)
          (C-L*CHANGE)
          (WATER-C)
          (CLUTCH-S)
          (S-K-UNIT NG)
          (PLUG CONTAMINATE)
          (PLUG-C (1 3))
          (KAITEN DOWN)
          (AIR-CLEANER CONTAMINATE)
          (PLUG UM)
          (O-Q NG)
          (C-AS S)
          (L COMP)
          (CHOKE)

*****MATCHED-RULE:R140*****

**********************************************
DATA-BASE:(** (CARBON))
          (** (PLUG CONTAMINATED))
          (** (ES AT SOON))
          (NOISE)
          (C-L*CHANGE)
          (WATER-C)
          (CLUTCH-S)
          (S-K-UNIT NG)
          (PLUG CONTAMINATE)
          (PLUG-C (1 3))
          (KAITEN DOWN)
          (AIR-CLEANER CONTAMINATE)
          (PLUG UM)
          (O-Q NG)
          (C-AS S)
          (L COMP)
          (CHOKE)

******-MESSAGE 1******

(PLUG IS CONTAMINATED. PLEASE CLEAN-UP YOUR ALL SPARK-PLUGS)

******-MESSAGE 2******

(PLUG NO YOGORE GA GEIIN DESU)

******-MESSAGE 3******

(CARBON WA OIL NO QUALITY GA WARUIKARADA.)

********GOOD-BYE********
>
```

図3　故障診断エキスパートシステムの実行例

4．考　察

今回使用例としたオートバイ故障診断のケースは，対象システムが小さいこと，ルール数が少ないのでシステムよりの質問がやや多いことなどの問題点はあるが，その結果得られた実行例からわかるように，開発ツールとしての有効性は満足できるものである。しかし，さらに複雑なシステムに関するエキスパートシステムを生成して，その測定結果から本開発ツールの機能を調査することは今後の課題である。

前にも述べた通り本システムはCMS上のLisp F3[7]にて記述されたが，上述の生成されたエキスパートシステムをIBM4361上のUtilisp(CMS版)[8]にて実行させてみたところそのまま実行可能であった。前者のLispがlnterlisp系であり後者がMaclisp系であることを考えると，実現の過程で考慮した移植性は充分達成されている。

コード量については生成された診断システムが全体で約1,600行であり，診断対象システムが小さい点を考えても満足できる範囲である。

結　言

手近に利用可能なエキスパートシステム開発用ツールを純粋プロダクションシステムをベースに試作した。簡単な対象に対して診断システムを生成し実行した結果から，本研究におけるアプローチは充分有効なものと考えられ，さらに大きな対象に対する診断システムの生成にもとづく評価が望まれる。

謝　辞

プログラムの作製について浦田知浩氏（現，東京コンピュータサービス）に協力願った。ここに謝意を表します。

引用文献

1）Lindsay, R. K., B. G. Buchanan, E. A. Feigenbaum and J. Lederberg: Applications of Artificial Intelligence for Organic Chemistry — The DENDRAL Project, McGraw-Hill, New York (1980)

2）Shortliffe, E. H.: Computer-Based Medical Consultations: MYCIN, Elsevier, New York (1976)

3）Duda, R., J. Gaschnig and P. E. Hart: Expert Systems in the Micro-Electronic Age, pp153, Edinburgh Univ. Pr., Edinburgh (1979)

4）Forgy, C. L.: OPS5 User's Manual, CMU-CS-81-135, Carnegie-Mellon Univ. (1981)

5）辻井：情報処理，**20**，735（1979）

6）斉藤，溝口：知的情報処理の設計，コロナ社，東京（1982）

7）Nordstrom, M.: LISP F3 User's Guide, Uppsala Univ., Uppsala (1978)

8）Chikayama, T.: Utilisp Manual, Tokyo Univ. (1981)

（受理　昭和60年1月30日）