

# 移植性を意識した basic-PROLOG システム

市 川 新

## I はじめに

人工知能の研究は情報科学の中核的テーマである。研究の目的は人間のもつ知的能力を機械知能として実現することにある。逆にいえば、人間の知能を用する情報処理を機械ができるとすれば、その機械は人工知能をもっているとされる。この研究には、機械知能を実現しようという具体的課題とともに、人間の知能とは何かという哲学上の課題も含まれており、それぞれの立場からの接近が試みられている。

人間の情報処理は情報行動に基づくものである。情報科学はそこにみられる情報現象の解明と情報理論の構築をしなければならない。ところが、これらはその時点で人間が創りだせる情報処理方式を通してのみできうる。人工知能の研究は、現象の解明、理論の構築および方式の開発の諸相の一体化が強く要請される。

一般の問題解決プログラムの研究は、経営管理における意思決定の場面において、この研究視座を実践している代表的なものである<sup>1)</sup>。前述のように、この種の研究の成否はその時点で実現されている情報処理方式を陽に示す人工言語、すなわち、プログラム言語に依存している。

従来、人工知能プログラム言語としては LISP が広く採用されてきた。この言語は記号処理を目的として開発されたものである<sup>2)</sup>。この言語で人間の知的情報処理方式を記述するには、論理表現あるいは推論機構自体も合わせて記述することになる。

一方、PROLOG は、述語論理に基づく言語

であり、言語処理システム自体が推論機構を内蔵している。この特徴により、LISP よりも高水準な人工知能プログラム言語といわれている。比較的知識が豊富であり、知識表現あるいは知識構造の形式化が進んでいる分野に限れば、人間の情報処理ないし知識情報処理を記述するプログラム言語として、PROLOG への期待が大きい。

PROLOG は、最初の処理系が 1970 年にマルセイユ大学で稼動したといわれ、それ以後、多くの大学で研究開発されている。わが国では東京大学で稼動している処理系が良く知られている。また、第 5 世代コンピュータの核言語としても研究されている<sup>3)</sup>。最近では、マイクロプロセッサへの PROLOG 移植が研究されている<sup>4,5)</sup>。

また、それぞれの専門分野での知識を記述する言語としての PROLOG の可能性も研究されている。例えば、歴史教育の対象に関する知識表現<sup>6)</sup>、裁判における法律の適用に関する知識表現<sup>7)</sup>、また、企業経営分析のための知識表現<sup>8)</sup> 等がある。これにともなって、言語処理系として求められる機能と適用分野の知識表現との整合性が新しい課題となる。

本稿では、この観点から、移植性と拡張性な

1) H. A. Simon: "The New Science of Management Decision" Prentice-Hall (1977).

2) J. McCarthy et al.: "LISP 1.5 Programmer's Manual" MIT Press (1962).

3) 淵一博: "述語論理型言語" 情報処理, 22 巻, 6 号 (1981).

4) 松田秀雄, 田中克己: "パソコン用 Micro Prolog の試作" bit, 14 巻, 7 号 (1982).

5) C. A. Sammut and R. A. Sammut: "The Implementation of UNSW-PROLOG" The Australian Computer Journal, Vol. 15, No. 2 (1983);

6) R. Ennals: "Artificial Intelligence: Applications to Logical Reasoning and Historical Research" Ellis Horwood (1985).

7) 池田純一: "人工知能言語による法律の解釈と適用" 日経コンピュータ, 73~4 号 (1984).

8) 後藤玉夫: "PROLOG による経営分析" ASCII, 9 巻, 12 号 (1985).

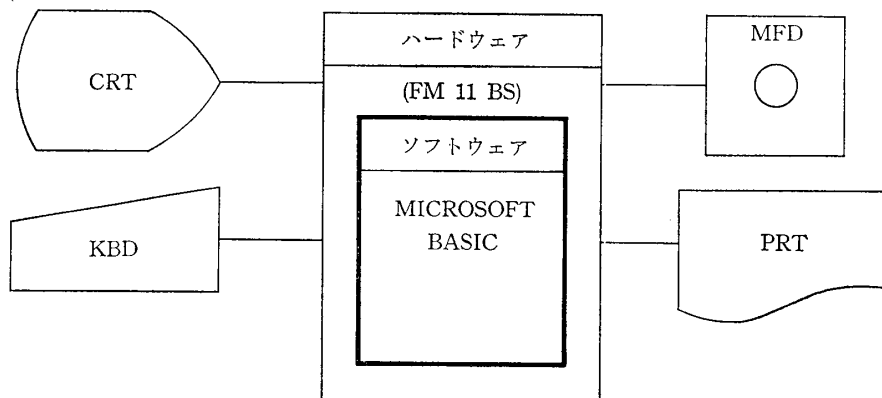


図1 処理系のシステム環境

いし適用分野特有の知識表現への特化を意識した PROLOG 処理系について述べる。移植性を実現するためにマイクロソフト系 BASIC 上にインプリメントする。これにより広く流通しているパソコンで PROLOG 処理系が得られる。また、適用分野への特化は PROLOG 処理系特有の変項代入、代入合成、統一化および非決定的処理の諸機能を陽にプログラム構造化することによって得られる。以後、この PROLOG 処理系を basic-PROLOG (b-PROLOG と略記) と呼ぶ。

## II b-PROLOG のシステム環境

b-PROLOG を実現したシステム環境を図1に示す。ハードウェア構成として Fujitsu Micro 11 BS を採用しており、現時点におけるパソコンの標準機器構成というべきものである。本稿で述べる b-PROLOG ではこれ以外の機器を必要としない。

一方、ソフトウェア環境はマイクロソフト系 BASIC に従属している。特に一連の文字列操作関数群に積極的に依存させており、この結果、非マイクロソフト系 BASIC との移植性は損なわれる。しかし、これら文字列操作で提供される機能によって、b-PROLOG システムを比較的単純化できるとともに、システムを記述したプログラムの可読性もよい。

PROLOG のプログラムは再帰的であり、計算過程において多量の記憶域を消費する可能性がある。b-PROLOG でも一時的記憶域を環境へ

要求するので、必要とする記憶容量を仮想的に確保するためにガベッジ・コレクタ機構の実現方法が問題となる。マイクロソフト系 BASIC では、文字領域中の廃品文字列の自動回収再生機構をシステム内蔵としている。これと PROLOG の内部表現との整合性が得られれば、PROLOG システム自体でのガベッジ・コレクタ機構の記述は不用である。したがって、b-PROLOG では、処理速度を犠牲にすることになるが、PROLOG の内部表現を文字(列)として展開している。

## III 述語論理式の表現

述語論理学は特定の対象間の特定の関係を問題にするのではない。ある集合に属する対象について、対象のとらえ方であるそれらの性質や関係を明瞭に表現する論理系が述語論理である。

一般に述語論理学では  $K_{xa}$  や  $R_y$  などに表現されたものを命題関数という。特定の対象を示す  $a$  等を定項、不定の対象を示す  $x$  や  $y$  を変項、 $K$  や  $R$  を述語という。命題関数はその変項に定項を代入すると命題となり、その命題の真値を決定できる。ここで、変項によって表わすものが対象のみに限られる場合を1階述語論理とっており、論理式はつぎのような標準形に変換できる。

$$P_1 \vee P_2 \vee \dots \vee P_n \leftarrow Q_1 \wedge Q_2 \wedge \dots \wedge Q_m$$

左辺の  $P_i$  を1個に限定した論理系を特にホーン論理と呼んでいる。すなわち、PROLOG でプログラムを記述することは、ホーン節で記

述することに等しい。

b-PROLOG ではホーン節の表記をつぎの形式にしている。

①  $+P, Q_1, Q_2, \dots, Q_n$

②  $+P$

③  $, Q_1, Q_2, \dots, Q_n$

ただし、 $P, Q_1, \dots, Q_n$  は項である。また、 $P$ をホーン節の頭部、 $Q_1, \dots, Q_n$ を本体と呼ぶ。さらに、ホーン節が頭部を含まない場合、本体はゴールの集りとなり、これをゴール節と呼ぶ。

項はアトムまたは複合項である。複合項はつぎの形にしている。

$f(t_1, t_2, \dots, t_n)$

ここで、 $f$ はシンボルまたは変項であり、述語名または関数名となる。

アトムは、他とまぎらわしくない任意の文字列のシンボルと数値である。数値は数字による文字列と区別するために接頭辞（'）をとる。すなわち、つぎのものは数値アトムとなる。

'-123 '4.56 '-2.34 E-02

変項は接頭辞(\*)のあとにシンボルが続いたものである。すなわち、つぎのものは変項となる。

\*x \*who \*what \*which

また、複合項の引数として現われる項にはつぎの項の並び（リスト）も許す。項の並びは、

[項1, 項2, …, 項 $n$ ]

の形式であり、項と項との対は

[項1|項2|[…]] または [項1|項2]

の形式であり、両者は等価である。

#### IV b-PROLOG の文法

b-PROLOG において入力されるものはつぎの3種類である。

① コマンド

② ホーン節

③ ゴール節

コマンドは、b-PROLOG の評価時におけるモードを指示するもので、表1にコマンド一覧を示す。コマンドにはつぎの形がある。

コマンド名 または コマンド名 (パラメー

表1 basic-PROLOG・コマンド一覧

コマンド名	パラメータ	説明
save	ファイル名	プログラム（ホーン節群）をファイルに書き出す
load	ファイル名	ファイルからプログラムを読み込み、付け加える
list	なし	最初のホーン節から20番目までを表示する
	節番号	節番号のホーン節から20番目までを表示する
llist	なし	プログラム・リストを印字する
new	なし	プログラム領域を初期化する
trace	0	評価過程をトレースしない
	1	探索の深さのトレースを表示する
	2	評価の結果を印字する
	3	統一化に成功したホーン節を表示する
hard	なし	評価過程の全ゴールのトレースを印字する
	なし	画面のハード・コピーを印字する
free	なし	ガベッジ・コレクションの起動及び空き領域の表示をする
quit	なし	basic-PROLOG の実行を終了する

タ)

コマンドの詳細については省略するが、trace コマンドの概要はつぎのとおりである。

trace コマンドは、PROLOG プログラミングにおいて、デバッグの困難さを軽減する。PROLOG のデバッグ支援については定着した方法が確立されていないが、b-PROLOG では trace コマンドによりゴール節の述語の評価過程を明示することにより対応している。

すなわち、trace(0) は評価過程を明示しない作用をもたらす。trace(1) は探索・推論の深さを動的に表示する。trace(2) は、さらに、評価の結果を印字する。trace(3) は、さらに、統一化に成功したホーン節を印字する。trace(4) は、さらに、評価過程における全てのゴール節を印字する。

PROLOG のプログラムはホーン節の集まりからなる。b-PROLOG におけるホーン節の形式は前述したとおりである。これをプログラミングという行為からみると、1つのホーン節は、

表 2 basic-PROLOG の組み込み述語

組み込み述語名	形 式		説 明
入出力述語	入 力	inp (変項)	項は変項である。入力要求をおこす。項に入力結果が代入される。
	出 力	prt (項1, 項2, ..., 項3)	項は定項または変項である。定項はそのまま, 変項は代入されているものが出力表示される。
制 御 述 語	strong cut	!	評価中に後戻りが起きたとき, 直ちに, これを生成した述語自体を統一化失敗したとする。
	fail	fail	無条件に統一化失敗とし後戻りを起こさす。
比較演算述語	数値比較	eqn (項1, 項2) nen (項1, 項2) ltn (項1, 項2) len (項1, 項2) gtn (項1, 項2) gen (項1, 項2)	項は数値に限る。それぞれ順番に {=, <, <, <=, >, >=} の数値比較をする。得られた結果 (真または偽) が述語の値となる。
	文字比較	eqa (項1, 項2) nea (項1, 項2) lta (項1, 項2) lea (項1, 項2) gta (項1, 項2) gea (項1, 項2)	項は文字列に限る。それぞれ順番に {=, <, <, <=, >, >=} の文字の内部コード値による文字比較をする。得られた結果 (真または偽) が述語の値となる。
算術演算述語	加 算 減 算 乗 算 除 算 剰 余	add (項1, 項2, 項3) sub (項1, 項2, 項3) mul (項1, 項2, 項3) div (項1, 項2, 項3) mod (項1, 項2, 項3)	項1と項2は数値, 項3は変項に限る。項1と項2に指定された算術演算がなされ, その結果が項3に代入される。

その頭部の述語の定義についての部分であり, 主張と呼べる。また, ある述語に関する主張の全体は述語の定義と呼べる。

ホーン節の形式には, 特殊形式として, ホーン節群によるプログラムの編集形式がある。

#### 節番号 ホーン節

ここで, 節番号は既に登録されていて, 付番済のホーン節番号である。この入力形式により, 節番号で指定した位置のホーン節が登録され, 対応する登録済ホーン節とそれ以後の節番号は繰り下がる。また, 節番号のみの入力形式はその節番号をもつ登録済ホーン節が消除され, 以下の節番号が繰り上がる。

ゴール節は, 前述のように, ホーン節の頭部がない形式である。ゴール節が入力されるとただちに証価を開始する。これにより, 述語論理学でいうところの, 命題関数の変項に定項が代入され, 命題の真理値すなわち真であるか偽を決定するために解の探索をする。

## V b-PROLOG の組み込み述語

b-PROLOG は拡張ないし適用分野特有の知識表現への特化を意識した処理系である。この観点から, 組み込み述語の機能選択と述語の組み込みの容易性が b-PROLOG に求められる。本稿では, 神戸大学の研究グループによる Micro Prolog を参考にして組み込み述語を選択している<sup>4)</sup>。また, 有用なシステムとして最低限必要となる入出力述語も組んでいる。表2に b-PROLOG の組み込み述語の一覧を示す。これらの組み込み述語の基本形式はつぎのとおりである。

入出力述語には inp と prt がある。b-PROLOG では, 処理系が自動的にゴール節の変項に代入された定項を出力する機能をもっていない。処理系としてはゴール節の真理値である YES または NO を応答するだけである。したがって, 出力すべき変項の代入値を陽に指示するために prt 述語を提供する。さらに, prt は

定項をそのまま出力するので、出力様式を編集することができる。inp は入力装置に要求を起動する。

制御述語は評価途中で強制的に評価手順を制御する場合に使われる。strong cut は、評価中に後戻り（バックトラッキング）が起これば、それ以上のゴール節の導出を強制的に抑制する。具体的には、strong cut を含むホーン節をゴール節化した、該当述語自体を失敗させる。fail は無条件に評価の後戻りを起動する場合に使われる。

比較演算述語には数値比較述語と文字比較述語がある。それぞれ、eq (equal), ne (not equal), lt (less than), le (less than or equal), gt (greater than), ge (greater than or equal) の6述語がある。数値比較述語には述語名に接尾辞である n (numeric), 文字比較述語には a (alphabetical) を付ける。真理値は、2項に数値または文字（列）が代入されていて論理条件を満足するとき真となる。

算術演算術語は項1と項2が数値、項3が変項である場合に限り真理値は真となる。項3には項1と項2を四則演算した結果の値が代入される。

## VI 処理系のシステム構造

PROLOG 処理系では、変項とその代入項の環境内部表現と後戻り制御を中心とする推論機関の実現方法が重要な研究課題である。b-PROLOG では、前述のように、システム記述言語として採用したマイクロ・ソフト系 BASIC の文字（列）処理に従属させている。内部表現および推論機関の実現方法は外部表現形式をそのまま処理対象とし、中間言語方式は採用していない。

例えば、b-PROLOG では階乗計算の定義をつぎのように記述する。

```
+factorial ('0, '1)
+factorial (*X, *Y), sub (*X, '1, *X1)
                    , factorial (*X1, *Y1)
                    , mul (*X, *Y1, *Y)
```

ここで、2の階乗値を得るには、ゴール節  
, factorial ('2, \*Z)

を実行すればよい。b-PROLOG では \*Z の代入項を出力させるには「, factorial ('2, \*Z), prt (\*Z)」としなければならないが、ここでは直接関係ないので省略する。

さて、factorial の定義は2つの主張により成り立つ。第1の主張は、それぞれの第1項である '2 と '0 が一致しないので統一化は失敗となる。第2の主張とは、\*X'←'2 および \*Y'←\*Z の代入合成ができる。ここで、' は探索が深くなっていく過程を陽に示し、順次個数が増加する。また、変項の主張への局所性を示す内部表現でもある。

処理系は第2の主張の本体を新しいゴール節として内部生成し評価を開始する。

```
, sub ('2, '1, *X1')
```

を評価すると、これは組込み述語であるので直ちに \*X1'←'1 の代入合成が行われる。

```
, factorial ('1, *Y1')
```

をつぎに評価することになるが、自分自身である第2の主張と統一化するので再帰的呼出しとなる。\*X''←'1 および \*Y''←\*Y1' の代入合成ができる。

```
, sub ('1, '1, *X1'')
```

をつぎに評価することになり、\*X1''←'0を得る。

```
, factorial ('0, *Y1'')
```

の評価は、第1の主張と統一化できるので、\*Y1''←'1 を得る。

```
, mul ('1, '1, *Y'')
```

の評価は \*Y''←'1 となる。最後に残ったゴールが

```
, mul ('2, '1, *Z)
```

となり、\*Z←'2 とすることにより全てのゴール節の評価を終る。

実際の変項の内部表現は、' の代わりに接尾辞 .nn を付加している。ここで nn は ' の個数を示す。より詳細な内部表現の変形過程はコマンド trace(4) で知ることができる。

b-PROLOG では、これら推論機関の機能実現のために、図2に示す記憶構造を採用してい

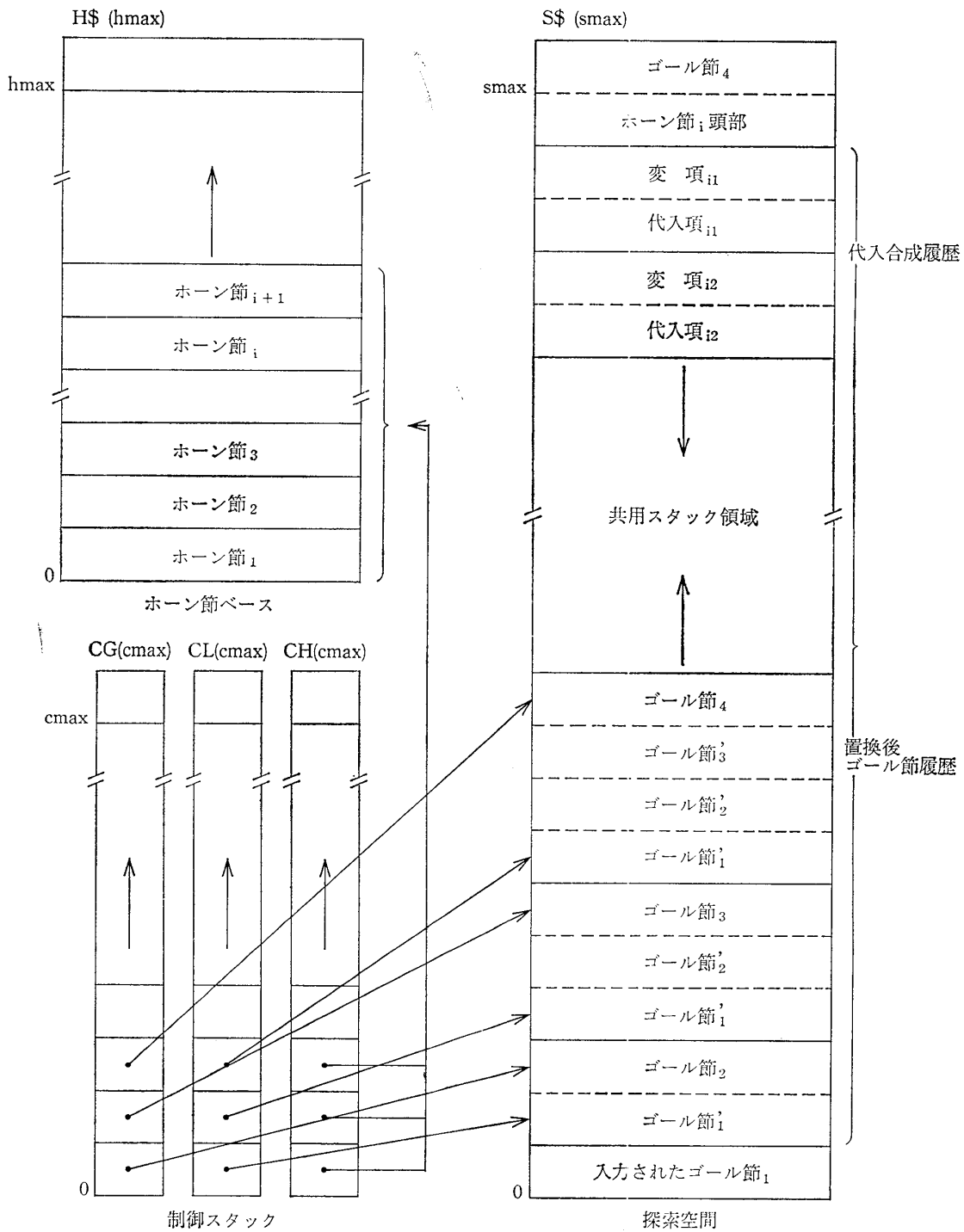


図2 b-PROLOG:メモリ・マップ

る。探索空間は、変項と代入項の対リストである代入合成履歴と評価過程で順次内部生成される置換後ゴール節履歴を記録する共用スタック構造となっている。制御スタックのCGはゴール節の位置、CLはそれぞれのゴール節の長さ、CHはそのゴール節を生成したホーン節の位置を記録している。

図3は b-PROLOG 処理系のシステム構造を示す。それぞれのブロック名と領域名は、付録に添付した原始プログラム上のサブルーチン名と変数名に対応している。より詳細な技術的事項については別稿に譲る<sup>9)</sup>。

<sup>9)</sup> 市川新: "Basic による PROLOG システム" 啓学出版 (近刊予定).

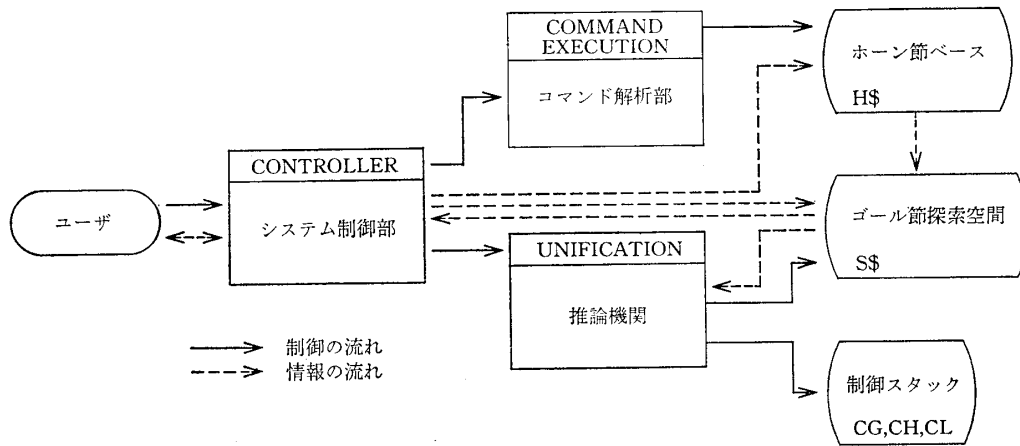


図3 b-PROLOG 処理系構造

```

===Welcome to basic-PROLOG===
load(ZU4)
list
1+append([*L]*X,*Y,[*L]*Z),append(*X,*Y,*Z)
2+append([],*Y,*Y)
3+sent(*X),np(*Y),vp(*Z),append(*Y,*Z,*X)
4+np(*X),det(*Y),noun(*Z),append(*Y,*Z,*X)
5+vp(*X),aux(*Y),verb(*Z),append(*Y,*Z,*X)
6+det([the])
7+noun([computer])
8+aux([will])
9+verb([die])
,sent([the,computer,will,die])

YES! ?=n
quit
    
```

図4 変形生成文法による構文解析例

## Ⅶ おわりに

最後に、b-PROLOG による簡単な使用事例を示す。図4は極めて簡単な英文法と辞書をPROLOGで記述したものである。これにより英文の構文解析を行える。それぞれの述語の定義はつぎのとおりである。

- append→第1項と第2項を付加えて第3項とする
- sent→名詞句に動詞句を付加える文である
- np→限定詞に名詞を付加えると名詞句である
- vp→助動詞に動詞を付加えると動詞句である
- det→限定詞である
- noun→名詞である
- aux→助動詞である

verb→動詞である

以上を定義しておくこと、「the computer will die」を構文解析できる。システムは真 (YES) と応答する。

さて、b-PROLOGで改良すべき点は、ホーン節ベースのファイル化と検索の効率化である。ホーン節ベース管理の手法が相対的に弱いので、知識情報処理の教育研究用の処理系としては不十分である。しかし、原始プログラムの大きさは約 10 KB の小型版であり、標準的パソコンであればホーン節ベースと探索空間を合わせて最大 65 KB まで確保できる。したがって、適用する課題を選定すれば PROLOG プログラムを記述できる。

## 付 録 basic-PROLOG システム

以下のプログラムを個人として利用するほかは、著作権法上著作者に無断で使用することはできない。

```
1000 '===basic-PROLOG (v.1.0)===
1010 '
1020 'Arata Ichikawa
1030 'Faculty of Economics, Ryutsu Keizai University
1040 'Ryugasaki 301, Japan. Tel. (2976) 2-3251
1050 'Copyright (C) 1985 by Arata Ichikawa. All rights reserved.
1060 '
1070 '===TOP LEVEL===
1080 '
1090 'CONFIGURATION
1100 DEFINT A-Z: T=-1: F=0: MHC=100: MCS=100: GA=300
1110 DIM H$(MHC), CG(MCS), CL(MCS), CH(MCS), S$(GA), AD$(2), DT#(2)
1120 HA=GA-1: MSS=GA-2: CSS=MSS: NHC=0: TS=F
1130 DEF FND(A1$)=A1$>="0" AND A1$<="9"
1140 MS1$="Horn space over.": MS2$="Input rejected."
1150 CLS: PRINT "===Welcome to basic-PROLOG===
1160 '
1170 'SYSTEM INPUT
1180 FOR M1=0 TO GA: S$(M1)="": NEXT: F1!=FRE(0)
1190 BEEP: LINE INPUT S$(0): GOSUB 3570: IF M1=0 THEN 1190
1200 B1$=LEFT$(S$(0),1):IF B1$="+" THEN GOSUB 3680: GOTO 1190
1210 IF FND(B1$) THEN GOSUB 3720: GOTO 1190
1220 IF B1$<>"," THEN GOSUB 3840: GOTO 1190
1230 IF TS>1 THEN LPRINT S$(0)
1240 '
1250 '===CONTROLLER===
1260 '
1270 'DOWN GOAL
1280 U=0: UL=0: VR=0: GOSUB 3090: CG(UL)=0: CL(UL)=1: CH(UL)=0
1290 '
1300 'UP GOAL
1310 AT=CG(UL): LG=CL(UL): IF S$(AT)<>"" THEN 1360
1320 IF LG>1 THEN CG(UL)=AT-1: CL(UL)=LG-1: GOTO 1310
1330 BEEP: PRINT: LINE INPUT "YES! ?=": B1$: IF TS THEN LPRINT "YES! ?=": B1$
1340 IF B1$<>"n" THEN 1620 ELSE 1180
1350 '
1360 AM=AT: GOSUB 2790: AT1=AM1: AT2=AM2
1370 G$=MID$(S$(AT),2,AT1-1)
1380 IF TS>3 THEN LPRINT "G": MID$(STR$(U),2): S$(AT) ELSE 1400
1390 IF LG>1 THEN FOR M1=1 TO LG-1: LPRINT S$(AT-M1): NEXT
1400 IF TS THEN PRINT ">";
1410 '
1420 'CONTROL OF BACKTRACKING
1430 IF G$="!" THEN GOSUB 3170: GOTO 1310 ELSE IF G$="fail" THEN 1620
1440 '
1450 'UP HORN
1460 IF LEN(G$)>>3 THEN 1540 ELSE B1$=MID$(S$(AT),2,AT1)
1470 M1=AT2-AT1-2: IF M1>0 THEN S$(GA)=MID$(S$(AT),AT1+2,M1) ELSE S$(GA)="
1480 IF G$="prt" THEN GOSUB 3390: GOTO 1310
1490 IF G$="inp" THEN 2240
1500 IF INSTR("add(sub(mul(div(mod(",B1$) THEN 2320
1510 IF INSTR("eqn(nen(ltn(len(gtn(gen(",B1$) THEN 2500
1520 IF INSTR("eda(nea(lta(lea(gta(gea(",B1$) THEN 2640
1530 '
1540 IF NHC=0 THEN 1620
1550 HD=CH(UL)
1560 IF HD=NHC THEN CH(UL)=0: GOTO 1620
1570 SWAP S$(HA),H$(HD): AM=HA: GOSUB 2790: HD1=AM1: HD2=AM2
1580 SWAP H$(HD),S$(HA): IF G$=MID$(H$(HD),2,HD1-1) THEN CH(UL)=HD+1: GOTO 1690
1590 HD=HD+1: GOTO 1560
1600 '
1610 'BACKTRACKING
1620 IF TS THEN PRINT "<";
```



```

1630 IF UL=0 THEN M$="NO!": GOSUB 3350: GOTO 1180
1640 U=U-1: UL=UL-1: FOR M1=CG(UL)+1 TO AT: S$(M1)="": NEXT: GOTO 1310
1650 '
1660 '===UNIFICATION===
1670 '
1680 'DISAGREEMENT
1690 S$(GA)=MID$(S$(AT),AT1+1,AT2-AT1): S$(HA)=MID$(H$(HD),HD1+1,HD2-HD1)
1700 U=U+1: UL=UL+1: VR=HA: GOSUB 3090
1710 IF UL>MCS THEN M$="Control space over.": GOSUB 3350: GOTO 1180
1720 '
1730 'MATCHING
1740 CSS=MSS
1750 M1=0
1760 M1=M1+1: IF S$(GA)=S$(HA) THEN 2050
1770 B1$=MID$(S$(GA),M1,1): B2$=MID$(S$(HA),M1,1)
1780 IF B1$=B2$ AND INSTR("<>.[ ]",B1$) THEN 1760
1790 IF B1$="*" OR B2$="*" THEN 1850
1800 IF B1$="" AND B2$="" THEN 1850
1810 IF B1$="|" OR B2$="|" THEN 1980
1820 IF B1$=B2$ THEN 1760
1830 U=U-1: UL=UL-1: CSS=MSS : GOTO 1540
1840 '
1850 S$(GA)=MID$(S$(GA),M1): S$(HA)=MID$(S$(HA),M1)
1860 TM=GA: GOSUB 2890: B3$=MID$(S$(GA),1,TM1): S$(GA)=MID$(S$(GA),TM1+1)
1870 TM=HA: GOSUB 2890: B4$=MID$(S$(HA),1,TM1): S$(HA)=MID$(S$(HA),TM1+1)
1880 IF B3$=B4$ THEN 1750
1890 IF B1$="*" OR B2$="*" THEN 1910
1900 IF VAL(MID$(B3$,2))=VAL(MID$(B4$,2)) THEN 1750 ELSE 1830
1910 IF CSS-2>AT THEN 1930
1920 M$="Unification space over.": GOSUB 3350: GOTO 1180
1930 IF B2$="*" THEN SG=HA ELSE SG=GA: SWAP B3$,B4$
1940 S$(CSS)=B4$: S$(CSS-1)=B3$: ST=CSS: GOSUB 3000
1950 CSS=CSS-2: GOTO 1750
1960 '
1970 'LIST TO DOT
1980 S$(GA)=MID$(S$(GA),M1+1): S$(HA)=MID$(S$(HA),M1+1)
1990 IF B1$="|" THEN M2=HA ELSE M2=GA
2000 IF B1$="]" OR B2$="]" THEN S$(M2)="["]+S$(M2): GOTO 1750
2010 S$(M2)="["+S$(M2): TM=M2: GOSUB 2890
2020 S$(M2)=LEFT$(S$(M2),TM1)+"["+MID$(S$(M2),TM1+1): GOTO 1750
2030 '
2040 'MATCH
2050 M1=AT+LG+1: IF TS>2 THEN LPRINT "H": MID$(STR$(HD),2): H$(HD)
2060 IF M1>CSS-1 THEN M$="Search space over.": GOSUB 3350: GOTO 1180
2070 S$(M1)=MID$(H$(HD),HD2+1): VR=M1: GOSUB 3090
2080 S$(M1-1)=MID$(S$(AT),AT2+1)
2090 IF LG<>1 THEN FOR M2=1 TO LG-1: S$(M1-1-M2)=S$(AT-M2): NEXT
2100 CH(UL)=0: IF S$(M1-1)<>"" THEN CG(UL)=M1: CL(UL)=LG+1: GOTO 2150
2110 S$(M1-1)=S$(M1): CG(UL)=M1-1: CL(UL)=LG
2120 '
2130 '===SUBSTITUTION===
2140 '
2150 IF CSS=MSS THEN GOTO 1310
2160 FOR ST=MSS TO CSS-2 STEP -2
2170 FOR SG=CG(UL) TO CG(UL)-CL(UL)+1 STEP -1: GOSUB 3000: NEXT
2180 S$(ST)="": S$(ST-1)="
2190 NEXT: CSS=MSS: GOTO 1310
2200 '
2210 '===BUILT-IN PREDICATES===
2220 '
2230 'inp( )
2240 IF MID$(S$(GA),1,1)<>"*" THEN 1620
2250 TM=GA: GOSUB 2890: S$(CSS)=LEFT$(S$(GA),TM1)
2260 BEEP: LINE INPUT "? "; B1$: M1=LEN(B1$): IF M1<1 THEN 2290
2270 IF INSTR(B1$," ") THEN 2290
2280 S$(CSS-1)=B1$: CSS=CSS-2: S$(AT)=MID$(S$(AT),AT2+1): GOTO 2150
2290 PRINT MS2$: GOTO 2260

```

```
2300 .
2310 'arithmetic +-*/*..
2320 FOR M2=0 TO 2: M1=F
2330 IF M2<2 AND MID$(S$(GA),1,1)<>"'" THEN M1=T: M2=2: GOTO 2370
2340 IF M2=2 AND MID$(S$(GA),1,1)<>"*" THEN M1=T: M2=2: GOTO 2370
2350 M3=INSTR(S$(GA),","): IF M2<2 THEN DT#(M2)=VAL(MID$(S$(GA),2,M3-2))
2360 IF M2=2 THEN AD$(2)=S$(GA) ELSE S$(GA)=MID$(S$(GA),M3+1)
2370 NEXT: IF M1 THEN 1620
2380 IF G$="add" THEN DT#(2)=DT#(0)+DT#(1): GOTO 2430
2390 IF G$="sub" THEN DT#(2)=DT#(0)-DT#(1): GOTO 2430
2400 IF G$="mul" THEN DT#(2)=DT#(0)*DT#(1): GOTO 2430
2410 IF G$="div" THEN DT#(2)=DT#(0)/DT#(1): GOTO 2430
2420 IF G$="mod" THEN DT#(2)=DT#(0) MOD DT#(1): GOTO 2430 ELSE 1620
2430 B1$=STR$(DT#(2)): M1=1
2440 M1=INSTR(M1,B1$," ")
2450 IF M1 THEN B1$=LEFT$(B1$,M1-1)+MID$(B1$,M1+1): GOTO 2440
2460 S$(CSS)=AD$(2): S$(CSS-1)=""+B1$: CSS=CSS-2: S$(AT)=MID$(S$(AT),AT2+1)
2470 GOTO 2150
2480 '
2490 'numeric <>=..
2500 FOR M2=0 TO 1: M1=F
2510 IF MID$(S$(GA),1,1)<>"'" THEN M1=T: M2=1: GOTO 2510
2520 M3=INSTR(S$(GA),","): IF M3=0 THEN M3=LEN(S$(GA))+1
2530 DT#(M2)=VAL(MID$(S$(GA),2,M3-2)): S$(GA)=MID$(S$(GA),M3+1)
2540 NEXT: IF M1 THEN 1620 ELSE S$(AT)=MID$(S$(AT),AT2+1)
2550 IF G$="eqn" THEN IF DT#(0)=DT#(1) THEN 1310 ELSE 1620
2560 IF G$="nen" THEN IF DT#(0)<>DT#(1) THEN 1310 ELSE 1620
2570 IF G$="ltn" THEN IF DT#(0)<DT#(1) THEN 1310 ELSE 1620
2580 IF G$="len" THEN IF DT#(0)<=DT#(1) THEN 1310 ELSE 1620
2590 IF G$="gtn" THEN IF DT#(0)>DT#(1) THEN 1310 ELSE 1620
2600 IF G$="gen" THEN IF DT#(0)>=DT#(1) THEN 1310 ELSE 1620
2610 GOTO 1620
2620 '
2630 'alphabetic <>=..
2640 IF INSTR(S$(GA),"*") THEN 1620
2650 FOR M1=0 TO 1
2660 TM=GA: GOSUB 2890: AD$(M1)=LEFT$(S$(GA),TM1): S$(GA)=MID$(S$(GA),TM1+2)
2670 NEXT: S$(AT)=MID$(S$(AT),AT2+1)
2680 IF G$="eqa" THEN IF AD$(0)=AD$(1) THEN 1310 ELSE 1620
2690 IF G$="nea" THEN IF AD$(0)<>AD$(1) THEN 1310 ELSE 1620
2700 IF G$="lta" THEN IF AD$(0)<AD$(1) THEN 1310 ELSE 1620
2710 IF G$="lea" THEN IF AD$(0)<=AD$(1) THEN 1310 ELSE 1620
2720 IF G$="sta" THEN IF AD$(0)>AD$(1) THEN 1310 ELSE 1620
2730 IF G$="gea" THEN IF AD$(0)>=AD$(1) THEN 1310 ELSE 1620
2740 GOTO 1620
2750 '
2760 '===SUBROUTINES===
2770 '
2780 'UP ATOM
2790 N1=LEN(S$(AM)): AM1=N1: AM2=AM1: N2=0: IF N1<2 THEN RETURN
2800 FOR N3=2 TO N1: A1$=MID$(S$(AM),N3,1): N4=INSTR(",<")",A1$)
2810 IF N4=0 THEN 2850
2820 IF N4=1 THEN IF N2=0 THEN AM1=N3-1: AM2=AM1: N3=N1
2830 IF N4=2 THEN IF N2=0 THEN N2=1: AM1=N3-1 ELSE N2=N2+1
2840 IF N4=3 THEN IF N2<2 THEN AM2=N3: N3=N1 ELSE N2=N2-1
2850 NEXT: IF AM2<AM1 THEN AM1=AM2
2860 RETURN
2870 '
2880 'UP TERM
2890 N1=LEN(S$(TM)): N2=0: A1$=MID$(S$(TM),1,1): TM1=0
2900 IF A1$="[" THEN N4=1: A2$="[": A3$="]" ELSE N4=0: A2$="(": A3$=")"
2910 FOR N3=1 TO N1
2920 A1$=MID$(S$(TM),N3,1)
2930 IF N4=0 AND N2=0 AND INSTR(",l]",A1$) THEN TM1=N3-1: N3=N1
2940 IF A1$=A2$ THEN N2=N2+1
2950 IF A1$=A3$ THEN N2=N2-1: IF N2=0 THEN TM1=N3: N3=N1
2960 NEXT: IF TM1=0 THEN TM1=N1
```

```

2970 RETURN
2980 '
2990 'SUBSTITUTION-0
3000 N1=LEN(S$(ST)): N2=LEN(S$(ST-1)): IF N1=0 THEN RETURN
3010 N3=1
3020 N3=INSTR(N3,S$(SG),S$(ST)): IF N3=0 THEN RETURN
3030 A1$=MID$(S$(SG),N3+N1,1): IF A1$="" THEN 3050
3040 IF INSTR(",)-]I(",A1$)=0 THEN N3=N3+N1: GOTO 3020
3050 S$(SG)=LEFT$(S$(SG),N3-1)+S$(ST-1)+MID$(S$(SG),N3+N1)
3060 N3=N3+N2: GOTO 3020
3070 '
3080 'VARIABLE RENAMING
3090 N1=1
3100 N1=INSTR(N1,S$(VR),"*"): IF N1=0 THEN RETURN
3110 N1=N1+1: A1$=MID$(S$(VR),N1,1): IF A1$="" THEN 3130
3120 IF INSTR(",)-]I(",A1$)=0 THEN 3110
3130 S$(VR)=LEFT$(S$(VR),N1-1)+"."+MID$(STR$(U),2)+MID$(S$(VR),N1)
3140 N1=N1+3: GOTO 3100
3150 '
3160 'strong cut !
3170 S$(AT)=MID$(S$(AT),AT2+1): IF TS THEN PRINT "!";
3180 N1=0: N2=0: IF UL=0 THEN RETURN
3190 FOR N3=UL TO 0 STEP -1
3200 N4=0
3210 FOR N5=CG(N3)-CL(N3)+1 TO CG(N3)
3220 N6=0
3230 N6=N6+1: IF INSTR(N6,S$(N5),"") THEN N4=N4+1: GOTO 3230
3240 NEXT
3250 IF N3=UL THEN N2=N4
3260 IF N3<UL AND N2=N4 THEN N1=N3: N3=0
3270 NEXT
3280 N2=CG(N1)-CL(N1): N3=AT-LG
3290 FOR N4=1 TO LG: SWAP S$(N2+N4),S$(N3+N4): NEXT
3300 FOR N4=N2+LG+1 TO AT: S$(N4)="": NEXT
3310 CG(N1)=N2+LG: CL(N1)=LG: CH(N1)=CH(UL): UL=N1: RETURN
3320 RETURN
3330 '
3340 'PRINT RTN
3350 PRINT: PRINT M$: IF TS THEN LPRINT M$
3360 RETURN
3370 '
3380 'prt( , , )
3390 GOSUB 3450: N1=1
3400 N1=INSTR(N1,S$(GA),","): IF N1>0 THEN MID$(S$(GA),N1,1)=" ": GOTO 3400
3410 PRINT: PRINT S$(GA): IF TS THEN LPRINT S$(GA)
3420 S$(AT)=MID$(S$(AT),AT2+1): RETURN
3430 '
3440 'DOT TO LIST
3450 N1=1
3460 N1=INSTR(N1,S$(GA),"."): IF N1=0 THEN RETURN
3470 IF MID$(S$(GA),N1+1,1)<>"[" THEN N1=N1+2: GOTO 3460
3480 N2=LEN(S$(GA)): N3=0: N5=0
3490 FOR N4=N1+1 TO N2: A1$=MID$(S$(GA),N4,1)
3500 IF A1$="[" THEN N3=N3+1
3510 IF A1$="]" THEN N3=N3-1: IF N3=0 THEN N5=N4: N4=N2
3520 NEXT: IF N1+2=N5 THEN A2$="" ELSE A2$=","
3530 IF N5=0 THEN 3460
3540 S$(GA)=LEFT$(S$(GA),N1-1)+A2$+MID$(S$(GA),N1+2,N5-N1-2)+MID$(S$(GA),N5+1)
3550 GOTO 3460
3560 '
3570 'INPUT CHECK
3580 M1=0: N1=1
3590 N1=INSTR(N1,S$(O)," "): IF N1=0 GOTO 3610
3600 S$(O)=LEFT$(S$(O),N1-1)+MID$(S$(O),N1+1): GOTO 3590
3610 N2=0: N3=1
3620 N3=INSTR(N3,S$(O),"!"): IF N3 THEN N2=N2+1: N3=N3+1: GOTO 3620
3630 IF N2>=2 THEN 3650

```

```
3640 IF LEN(S$(0))>=1 THEN M1=1: RETURN
3650 PRINT MS2$: RETURN
3660 '
3670 'STORING HORN CLAUSES
3680 IF NHC>MHC THEN PRINT MS1$: RETURN
3690 H$(NHC)=S$(0): NHC=NHC+1: RETURN
3700 '
3710 'UPDATE HORN CLAUSES
3720 N1=0: N2=1
3730 A1$=MID$(S$(0),N2,1). IF NOT FND(A1$) THEN 3750
3740 N1=N1*10+VAL(A1$): N2=N2+1: GOTO 3730
3750 N1=N1-1: IF N1<0 OR N1>=NHC THEN PRINT MS2$: RETURN
3760 IF A1$="+" THEN 3780 ELSE IF A1$="" THEN 3770 ELSE PRINT MS2$: RETURN
3770 FOR N3=N1 TO NHC-2: SWAP H$(N3),H$(N3+1): NEXT: NHC=NHC-1: RETURN
3780 IF NHC>=MHC THEN PRINT MS1$: RETURN
3790 FOR N3=NHC TO N1+1 STEP -1: SWAP H$(N3),H$(N3-1): NEXT
3800 NHC=NHC+1: H$(N1)=MID$(S$(0),N2): RETURN
3810 '
3820 '===COMMAND EXECUTION===
3830 '
3840 AM=0: GOSUB 2790: A1$=LEFT$(S$(0),AM1)
3850 N1=AM2-AM1-2: IF N1>=1 THEN A2$=MID$(S$(0),AM1+2,N1) ELSE A2$=""
3860 IF A1$="quit" THEN END
3870 IF A1$="list" THEN IF NHC=0 THEN RETURN ELSE 3960
3880 IF A1$="llist" THEN IF NHC=0 THEN RETURN ELSE 3990
3890 IF A1$="new" THEN IF NHC THEN 4000 ELSE RETURN
3900 IF A1$="trace" THEN TS=VAL(A2$): RETURN
3910 IF A1$="load" THEN OPEN"I",#1, A2$: GOTO 4010
3920 IF A1$="save" THEN IF NHC THEN 4040 ELSE RETURN
3930 IF A1$="hard" THEN HARDC: RETURN
3940 IF A1$="free" THEN PRINT "Space size": STR$(FRE(0));".": RETURN
3950 PRINT "Command error.": RETURN
3960 N1=VAL(A2$)-1: IF N1<0 THEN N1=0 ELSE IF N1>=NHC THEN N1=NHC-20
3970 N2=N1+19: IF N2>NHC-1 THEN N2=NHC-1
3980 FOR N3=N1 TO N2: PRINT MID$(STR$(N3+1),2): H$(N3): NEXT: RETURN
3990 FOR N1=0 TO NHC-1: LPRINT MID$(STR$(N1+1),2): H$(N1): NEXT: RETURN
4000 FOR N1=0 TO NHC-1: H$(N1)="": NEXT : NHC=0: RETURN
4010 IF EOF(1) THEN CLOSE#1: RETURN
4020 IF NHC>MHC THEN PRINT MS1$: CLOSE#1: RETURN
4030 LINE INPUT#1, H$(NHC): NHC=NHC+1: GOTO 4010
4040 OPEN"O",#1, A2$: FOR N1=0 TO NHC-1: PRINT#1, H$(N1): NEXT: CLOSE#1: RETURN
```