新しいコンピュータ言語Prologの橋梁形式選定システムへの適用について

京都大学工学部　正員　白石成人
京都大学工学部　正員　松本勝
京都大学工学部　正員　古田均
京都大学大学院　学生員　○谷川浩司

1．まえがき

従来の橋梁計画・橋梁設計は、経験・直感などに基礎を置いており、意思決定の流れが不明確であり、全体としての一貫性に欠けているところが多くみられる。また、計画・設計を支援するための資料も体系的に整備されておらず、有効に活用されていないのが現状である。そこで筆者らは、橋梁の計画・設計のシステム化をめざし、「橋梁計画段階における形式選定システム」について研究を行い、その概要を参考文献（1)などにおいて報告してきた。本研究では、この「橋梁形式選定システム」を、コンピュータに支援されたより現実的なシステムに近づけることを目的とし、新しいコンピュータ言語Prologを用いて、システムの一部である「架設工法選定のルーチン」を、コンピュータ上に実現することを試みた。第５世代コンピュータ・プロジェクトの基本言語であるPrologを用いることにより、フローチャート・選定表などにより表現されたシステムを、簡潔かつ第三者にも理解しやすい形でプログラム化でき、高速で柔軟性に富んだ対話型システムの構築が可能になると思われる。以下、Prologの概要と適用例について述べ、得られた結論と今後の課題について記す。

2．Prologの概要[2),3)]

Prolog（Programing in Logic ）は、その名前が示すように述語論理型言語とも呼ばれ、計算機による定理証明の研究から生み出されたものであり、パターン・マッチングによる手続き呼び出し、再帰呼び出し、などの高度なプログラミング機能を基本機能として備えている。またプログラミングは、述語論理と呼ばれる事物間の関係に焦点をあてた形式言語により行われ、自然言語に最も近い表現力を有している。これは、従来のFORTRAN 、BASIC などが手続き中心の言語であったのと著しく異なり、この意味で、Prologは関係型言語、FORTRAN は手続き型言語とも呼ばれている。Prologの特徴および利点を以下に列挙する。

1)データとプログラムの区別がないこと。したがって、FORTRAN などによるプログラムでは、データはプログラムに記された定形的な加工手順に従って処理されたが、Prologでは非定形な情報処理が可能となる。

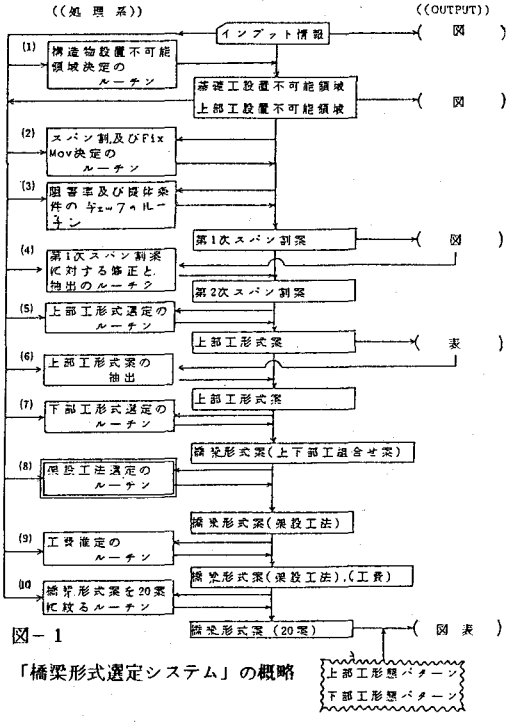
2)データとプログラムの区別がないため、プログラムの改変が容易であるとともに、プログラム自体もデータの加工手続きのための記述が不要となる分だけ簡潔となる。

3)三段論法的な推論を行うことができ、より人間の思考過程に近い形でプログラムが書ける。

4)自然言語表現に近いため、知識の表現あるいは思考の流れが第三者にとっても理解しやすく、プログラムの事後の修正が容易である。

5)強力なパターン・マッチング機能を有しているため、入力データと知識ベースとの照合が多いエキスパートシステム（後述）の開発に適している。

Prologの文（プログラム）には三種類あり、それを京都大学大型計算機センターで現在稼動中の、Prolog/KR 方式で表すと次のようになる。



図－１　「橋梁形式選定システム」の概略

a)事物とその関係についての規則を定義する。

(ASSERT (P) (Q_1) (Q_2) ・・・(Q_n)) n=1,2,3,・・・

「Q_1,Q_2,・・Q_nの全てが真ならば Pも真である」

「 Pを実行するには、Q_1,Q_2,・・Q_nを実行すればよい」

b)事物とその関係についての事実を宣言する。

(ASSERT (P))

「 Pが無条件に成り立つ」

C)事物とその関係について質問する。

(AND (Q_1) (Q_2)・・・(Q_n)) n=1,2,3,・・・

「Q_1,Q_2,・・Q_nを全て満足するものはあるか？」

ここで、P および Qは

(述語名 t_1 t_2・・・t_i) ; t_iは引数でi=1,2,・・

の形で表される。簡単な例として次のような推論を示す。

a)「ある橋が吊橋ならば、それはケーブルを持つ」

b)「関門橋は吊橋である」

c)「関門橋はケーブルを持つ」

これをProlog/KR を用いて実行すると次のようになる。

:(AS (HAVE *X CABLE)(SUSPENDED-BRIDGE *X))

; 規則 a) の形の文の入力 (AS はASSERTの略)

:(AS (SUSPENDED-BRIDGE KANMON-BRIDGE))

; 事実 b) の形の入力

:(HAVE KANMON-BRIDGE *X)

; 質問 c) の形で「関門橋の持つものは？」

これらの入力に対して、システムは

(HAVE KANMON-BRIDGE CABLE)

と応答し、「関門橋はケーブルを持つ」という意味の答を返す。なお上例で*Xは変数を表しており、Prologでは変数は、文が異なれば同じ名前であっても別々の対象を表す。

3．「架設工法選定システム」への適用[4),5)]

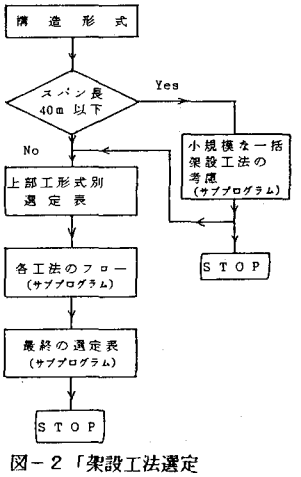
前述のPrologの特色を最大限に利用して、「橋梁形式選定システム」の一部である「架設工法選定システム」をPrologを用いてコンピュータ上へ実現した。本システムの概略を図－２に示す。プログラムの作成にあたっては、特に以下の諸点に注意した。

1)プログラムの内容が第三者にも理解しやすいよう、述語名、引数名の表現に注意した。

2)属性が同じであると考えられる知識を統合することにより、知識表現のモジュラリティーを高めた。

3)プログラムの今後の修正を容易にし、さらに思考の流れを明らかにするため、知識表現部とそれを用いての実行制御部（推論部）とを分離した。

4)システムの部分的利用が可能となるよう、各工法のフローそれぞれについて、サブプログラムの形でプログラムを組んだ。



図－２「架設工法選定システム」の概略

次に実際のプログラムを示す。図－２の内容をプログラム化したものがプログラム１である。これはシステム全体を実行させ、各工法のフロー用のサブプログラムを駆動させるためのメインプログラムとなっている。もし、ある工法の採用の可否だけを知りたい場合には、このプログラムをスキップし、各工法のサブプログラムだけを実行することもできる。

プログラム１

```
:(AS (KASETSU-START)(IF (SPAN YES)(SMALL-BLOCK)(CH-
 TABLE)))
:(AS (CH-TABLE)(PRINT "BRIDGE NO KEISIKI WA?")(PRIN
 T "1. SIMPLE PLATE OR BOX GIRDER BRIDGE") ・・・・
 (READ *S)(SELECT *S (1 (AND(BENT)(CABLE)・・・・・
                :
                :
```

さらに、より具体的なプログラムの例として、各工法のフローの一つである大ブロック工法用のフローについて説明を加える。大ブロック工法のフローを図－３に示す。プログラム２－Aは知識の表現部であり、図－３の点線枠に示すように、判断規則のうちよく似たものをひとまとめにして、(WATER) (TRANSPORT) などの代表名をつけることにより、知識表現のモジュラリティーを高めてある。例えば (WATER) という、桁下の水深などに関する３つの規則からなる知識は、一文で非常に簡潔に表現されている。また、述語名、引数名も (UND-BR WATER)=There is water under the bridge. (DREDGE OK) = Dredging is OK. などのように、実際の英文を簡略化したもので、自然言語表現に近い判読しやすいものに

条件		工法	選定表７ F・C	選定表８ 巻上げ台船機
桁下高	0 ～15m		○	○ ◎
	15～50m		◎	○ ×
	50m～		×	◎ ×
ONE-BLOCK の重量	0 ～2000t		◎	◎ ○
	2000～4000t		△	◎ ◎
	4000t ～		×	△ ◎
上部工形式	I 又BOX(単純)		◎	× ○
	I 又BOX(連続)		○	○ ○
	単純トラス		○	× ○
	連続又ゲルバートラス		○	○ ○
	ゲルバートラス(吊桁)		○	◎ ○
	斜 張 橋		○	△ ○
桁上空のクリアランス	有		○	○ ○
	無		×	○ △

表－１ 選定表７，８

なっている。さらに、(BL-HIGHT) (BL-STYLE) などは、表－１に示す選定表７，８をProlog/KR の組み込み述語 COND、SELECTを用い、さらに◎○△×をそれぞれ3,2,1,0点と得点化して示したものであるが得点やカテゴリーなどの変更が容易に行え、読みとりやすい表現となっている。

プログラム２－A（知識表現部）

```
:(AS (WATER)(AND (UND-BR WATER)(OR (WATER-DEP OK)(A
 ND (WATER-DEP NO)(DREDGE OK)))))
:(AS (WATER-WAY)(OR (EX-WW NO)(AND (EX-WW YES)(CONT
 ROL-WW OK))))
:(AS (TRANSPORT)(AND (OR (TRANS-BLOCK OK)(AND (TRAN
 S-BLOCK NO)(SPACE-FAB YES)(SHIPMENT-BAR OK))) (APP
 R-BAR OK)))
:(AS (FAB-ON-BARGE)(AND (SPACE-PARTS YES )(APPR-EN-
  BAR OK)(FAB-BAR OK)))
                    :
:(AS (BL-HIGHT)(PRINT "KETASITA NO TAKASA WA?(M)")
 (READ *X)(COND
 ((<= *H 15)(AND (= *FC 2)(= *WIN 2)(= *BAR 3)))
 ((> *H 50)(AND (= *FC 0)(= *WIN 3)(= *BAR 0)))
 ((TRUE)(AND (= *FC 3)(= *WIN 2)(= *BAR 0))))
 (SET BL-PT1 (*FC *WIN *BAR)))
                    :
:(AS (BL-STYLE)(PRINT "BRIDGE NO KEIS
 IKI WA?")(PRINT "1.SIMPLE PLATE OR
 BOX  GIRDER BRIDGE")(PRINT "2.CONTI
 ・・・  (READ *S)(SELECT *S (1
 (AND (= *FC 3)(= *WIN 0)(= *BAR 2)))
 (2 (AND (= *FC 2)(= *WIN 2)(= ・・・
                    :
```

次にプログラム２－Bは、２－Aで表現された知識を用いて、フロー（図－３）全体を実行する部分である。(BL-001)(BL-002)などは、図－３に示すように (WATER) (TRANSPORT)などの統合された知識の関係を示し、それらの実行の順序を指定している。この部分は、FORTRAN などによるプログラムとよく似た手続き的な表現となっているが、知識の統合とあいまって、実行の流れを追いやすい簡潔な表現となっている。また(BL-CALCU)は、選定表７，８の得点計算を行う部分である。Prologでは、異なる文の間での値の受け渡しは、

(SET A (3 2 4))→(A (*X *Y *Z))

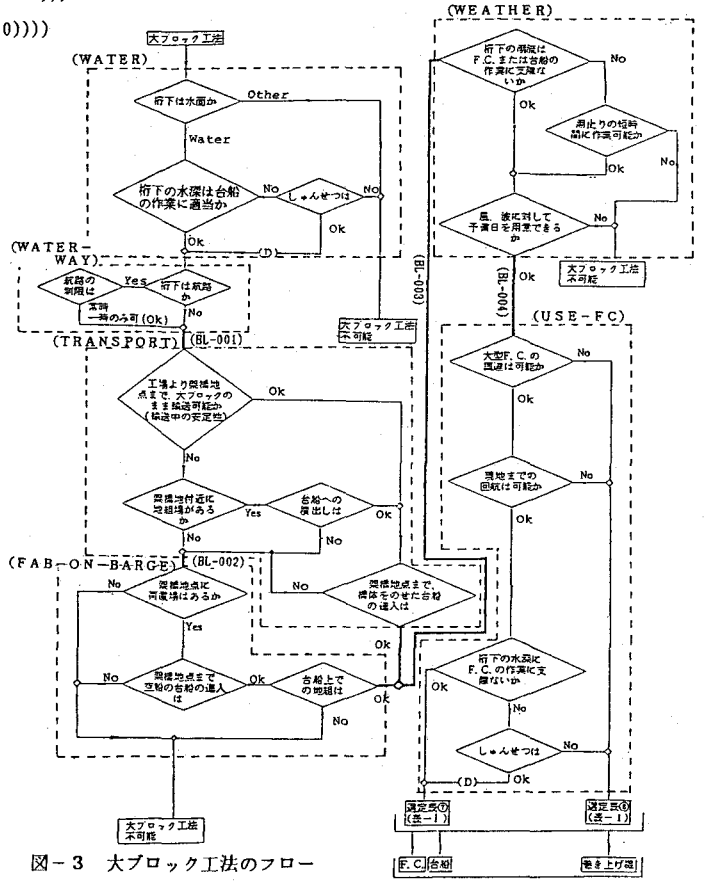
などの形で、さらに加・積などの算術計算も

(PLUS 1 2 *X) ，(TIMES 4 3 12)

の形で、それぞれパターン・マッチングにより行われるため、この計算部はなじみの薄い表現となっている。その他プログラム２－B内には、システム全体を実行するために必要な種々の手続きが表現されている。

プログラム２－B（実行制御部）

```
:(AS (BLOCK) (IF(AND(WATER)(WATER-WAY))(BL-001)
 (BLOCK-NO)))
:(AS (BL-001)(IF (TRANSPORT)(BL-003)(BL-002)))
:(AS (BL-002)(IF (FAB-ON-BARGE)(BL-003)(BLOCK-NO)))
:(AS (BL-003)(IF (WEATHER)(AND (BL-CALCU)(BL-004))
 (BLOCK-NO)))
:(AS (BL-004)(IF (USE-FC)(TABLE7)(TABLE8)) (BL-CHOI
 CE) (BL-CAUTION))
:(AS (BL-CALCU)(AND (BL-HIGHT)(BL-WEIGHT)(BL-STYLE)
 (BL-U-CL))(BL-PT1 (*FC1 *WIN1 *BAR1))(BL-PT2・・・
 (TIMES *FC1 *FC2 *FC3 *FC4 *FCR)(TIMES  ・・・・・
 (IF(= *FCR 0)(= *FCRE 0)(PLUS *FC1 *FC2 *FC3 *FC4
```



図－３　大ブロック工法のフロー

```
  *FCRE))(IF (= *WINR 0)(= *WINRE  ・・・・・・・・
  (SET BL-POINT (*FCRE *WINRE *BARRE)))
 :(AS (BL-CHOICE)(BL-POINT (*FC *WIN *BAR)) ・・・・
                    :
```

プログラム２－Ｃは、ユーザーに質問を発しそれに対する答を受ける入出力部である。Prologでは、変数の有効範囲はその一文だけであるため、入出力に際して種々の変数名を用意する必要はない。また、パターン・マッチングを基本にしているため、入出力用の書式指定も不要である。

プログラム２－Ｃ（入出力部）

```
:(AS (UND-BR *X)(PRINT "KETASHITA WA? WATER/OTHER")
  (READ *Y)(SET UND-BR *Y)(= *X *Y))
:(AS (WATER-DEP *X)(PRINT "KETASHITA NO SUISHIN WA
                    :
```

以上のように、プログラムを知識部、実行制御部（推論部）、入出力部に分離することは、プログラムの追加、削除、変更という操作性を向上させ、複雑なプログラムを、すっきりとした判読しやすいものとするのに貢献している。ところでこれらのプログラムを、FORTRAN を用いて作成するとどのようになるであろうか。おそらくは、上述のようなプログラムの三分割は困難であり、IF、GO TO などの制御命令が羅列し、多数の変数名が次々に出現する、複雑なプログラムとなるであろう。このため、変数の意味を理解し実行の流れをたどることは、非常に困難であると思われる。また入出力に際しては、書式の指定が必要となるので、プログラムの製作者にとって余分な負担になるであろう。さらにプログラムの一部変更は、他の部分へも大きな影響を及ぼし、例えば、知識やアルゴリズムの変更が、プログラム全体の書き換えにつながることも多いにありうると思われる。これらのことを総合的に判断すると、本システムにとってPrologはFORTRAN よりも有効なプログラミング言語になると考えられる。

付録に実際の TSS端末上での実行を示す。まず、(LOAD "BLOCK.TEXT")により、プログラム２－Ａ，Ｂ，Ｃの内容をProlog/KR に読み込ませる。次に(BLOCK) により実行を開始する。システムはユーザーに対して次々に質問を発し、それに対してユーザーは指定された形式で答える。この処理の過程で大ブロック工法の採用が不可能となった場合には、"DAI-BLOCK-KOHO WA SUBETE FUKA" と表示して実行を終了する。そうでない場合には、選定表部分の実行へと進み、使用機材それぞれについて合計点を表示する。それに対してユーザーは、その合計点を見ながら、自分の経験を折りまぜて使用機材を一つ選択する、という対話型の流れになっている。

４．まとめ

以上述べてきたように、Prologを用いたことにより、Prologに関する知識をあまり持たないような人でも、知識の内容、思考の流れが理解しやすい簡潔なプログラムを作ることができた。また、Prologの持つすぐれた基本機能を十分に活用することにより、応答時間の短い、柔軟性に富んだ対話型システムを実現することができた。これらのことよりPrologは、定性的な情報の判断を必要とされる「橋梁形式選定システム」にとって非常に有効なプログラミング言語になると考えられる。

ところで、本研究で対象とした「架設工法選定システム」で用いた判断知識は、マニュアルなどから得たいわば教科書的知識であり、思考の流れも、採用が不可能でないものをすべて選び出し最終的に選定表により２～３案に絞るというものである。このように、アルゴリズムが特にコンピュータ向きに編成されたものではないため、「橋梁形式選定システム」全体を考えた場合、実行時間の非常に長い効率の悪いシステムになると考えられる。このような問題の解決には、専門家の持つ知識・判断法の有効な活用を重視し、コンピュータ向けに設計されたエキスパートシステムの利用が有効であると考えられ、現在研究中である。

付録　（ TSS上での実行）

```
:(LOAD "BLOCK.TEXT")
:(BLOCK)
"KETASHITA WA? WATER/OTHER"
-:WATER
"KETASHITA NO SUISHIN WA BARGE NO SAGYOU NI
TEKITOU KA? OK/N"
-:OK
                    :
"TABLE7 NI YORU TOKUTEN WA
 1. FC NI YORU DAI-BLOCK-KOHO = 8
 2. WINCH NI YORU DAI-BLOCK-KOHO =10
 3. BARGE NI YORU DAI-BLOCK-KOHO WA FUKA"
"UENO TOKUTEN WO SANKOU NI SITE KOHO WO SENTAKU SI
BANGOU WO INPUT SITE KUDASAI"
-:2
"WINCH NI YORU DAI-BLOCK-KOHO WO SENTAKU"
```

（参考文献）

1)白石、松本、谷川:"橋梁計画段階における形式選定システムについて",土木学会第38回年次学術講演会，1983
2)平野:"プログラミング言語Prolog",京都大学大型計算機センター公報,vol.17,No.3,1984
3)中村訳:"Prologプログラミング",マイクロソフトウエア社，1983
4)Nakashima:"Prolog/KR User's Manual",METR82-4
5)福田:"Prologを用いた自動車エンジンの故障診断システムの試作",計測自動制御学会第 2回知識工学シンポジウム，1984