TR-170

Programming in Modal Logic

榊原康文(富士通)

April, 1986

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191~5
Telex ICOT J32964

Programming in Modal Logic

柳原　恵文
富士通㈱国際情報社会科学研究所

1.　導入

近年様々な論理プログラミングが提案され、開発されている。その代表格であるのが一階述語論理（正確にはその部分系であるホーン論理）に基づくPROLOGによるプログラミングである。しかし、そもそも一階述語論理は、何処でも何時でも表明される式の真偽は変わらず一定である事柄を対象としており、その記述される世界は一つで平坦なものである。これは数学などの超空間的・超時間的な分野の記述・表現には適切であるが、一般のプログラミングの対象には不便なことが多い。このように、一階述語論理に欠けるものは、空間的構造であり、状態変化・時間の推移であり、ひいては多義性である。これらを克服するため、PROLOGに様々な制御機構や構造を加え、改良する試みが研究されている（[2, 3, 4, 5, 6]）。一方、論理学の分野でも非標準論理と総称される一階述語論理と異なる論理系が研究されている。特に様相論理は、その(Kripke による）意味論において多世界モデルを導入し、必然・可能オペレータのシンタックスプリミティブと共に、空間的・時間的表現そしてそれによる多義性を扱おうとしている。今回、この様相論理の多世界モデルの概念をそのまま素直に取り込み、その上で行う論理プログラミングを提案する。これは、従来の論理プログラミングの構造化・モジュール化の研究と比べて、非常に単純な拡張でありかつ柔軟な表現力を持つ。さらにその背景には、様相論理の多世界モデル論という厳密な形式的意味論が存在する。そしてその応用として、知識の階層や属性の継承、知識の共有などの知識表現への適用、またオブジェクト指向などのモジュラー・プログラミングがいかに表現され得るかを議論し、また他の類似の研究との比較を行う。

2.　様相論理（[1]）

様相論理は、必然性，偶然性あるいは可能性という様相概念を扱う論理である。そしてその意味論は、可能世界に基づく多世界意味論である。すなわち、意味解釈において、我々は現実の世界だけを問題にするのではなく、様々な可能世界を想定する。現実の真理を決めるためには、現実の世界ひとつを考えれば十分なのであるが、これに対して必然的な真理を定めるためには、ひとつの世界を考察するだけでは不十分であって、与えられた（現実の）世界からみて＂可能な世界すべて＂を考えねばならないのである。

そこで意味解釈における様相論理のモデルは、述語論理のモデルに２つの要素を加えることによって得られる。１つは可能世界の集合W。もう１つはWの成員間に成立する＂相対的可能性の関係＂Rである。ある世界w_1がw_2に相対的に可能であることは「$w_1 R w_2$」と書かれるが、この関係に基づいて様相命題の真理値が決められる。このモデルにおいて、Aが現実の世界wで真ならAは現実に真である。またwRw'が成立するすべてのw'(つまり現実の世界からみて可能なすべての世界）でAが真であれば、□Aは真、すなわちAは必然的に真である。さらに、wRw'となるようなw'が少なくとも１つあって、Aがw'で真ならば、◇Aは真、すなわちAは可能である。

このように相対的可能性の関係Rを用いてモデル理論を展開しておくと、Rの条件をいろいろ変えることにより、種々の構造を持つ様相概念が扱える。例えば、Rに課せられる条件に、反射性，対称性，推移性などを組み合わせて考えることにより、T，S４，S５タイプのモデルによる様相論理系が得られる。そこで論理プログラミングにこのモデル論的解釈を導入し、可能世界の集合Wの成員を１つの閉じた世界（モジュール）として捉え、フラットな述語論理による論理プログラミングに構造を入れる。そしてWの世界間に成立する相対的可能性の関係を定義する構文を用意することにより、世界間の構造関係，階層性やモジュール化，局所化などの知識表現やプログラミング方法論における重要な概念が表現できるようになる。次節でこれを実現する様相論理プログラミング(Programming in Modal Logic)について紹介する。

3. Programming in Modal Logic

3.1. 言語

－構文－

〔プログラム〕

・$t_1,\ldots,t_n$ を項、pをn引数述語、lを変数とするとき、

$p(t_1,\ldots,t_n)$, $\Box p(t_1,\ldots,t_n)$, $\Diamond(l)p(t_1,\ldots,t_n)$, not p

はアトム。

・$A,A_1,\ldots,A_n$ をアトムとするとき、

$A \leftarrow A_1,\ldots,A_n$

は節。

・プログラムは、節の有限集合。

〔世界(world)〕

・「世界」の定義は、世界名とその世界のプログラムを定義する。一つの世界は、

```
world　世界名　of
  プログラム（節の有限集合）
fo.
```

の形をしている。

各世界でのプログラムの定義は、その世界で成り立つ属性や関係を記述する公理と考えることもできる。

〔関係(relation)〕

・「関係」の定義は、世界間の相対的可能性の関係を定義する。ある世界w」がw_iに相対的に可能であることを $re(w_i, w_j)$ と書く。さらに、関係間の関係を節で書くこともできる。これによって関係 re に課せられる条件、対称性・推移性などが記述できる。

・t_1,t_2 を世界名または変数とするとき、

$re(t_1,t_2)$ は関係アトム。

・$R,R_1,\ldots,R_n$ を関係アトムとするとき、

$R \leftarrow R_1,\ldots,R_n$ は関係節。

・関係は、関係節の有限集合であり、

```
relation of
  関係節の有限集合
fo.
```

の形をしている。

(例)

```
relation of
  re(w1,w2);
  re(w2,w3);
  re(X,Y)←re(X,Z),re(Z,Y)
fo.
```

〔様相論理プログラム〕

様相論理プログラムは、複数の世界の記述と一つの（それらの世界間の）関係の記述から成り、

```
relation of
  関係節の有限集合
fo.
world　世界名　of
  プログラム（節の有限集合）
fo.
  ・・・
world　世界名　of
  プログラム（節の有限集合）
fo.
```

の形をしている。

3.2. 意味

ここでは3.1.節で定義された構文に対して、様相記号のモデル論的解釈に基づき、様相論理をプログラミング言語（Programming in Modal Logic）と解釈する手続き的解釈を与えることを目的とする。

〔手続き的解釈〕

・様相オペレータの付かないアトム・節の意味は、通常の論理プログラミングにおける意味と同じである。

・様相オペレータの解釈：

様相論理の宣言的（モデル論的）解釈においては、□Aがある世界wで真であるということは、Aがwからみて可能なすべての世界で真であり、◇Aがwで真であるとは、wからみて可能な世界でAが真となる世界が存在する、と解釈される。一方、論理プログラミングでは、記述された論理式は、真となるものを表明していると考え、それを事実として加えられた公理であると解釈する。そこで様相論理プログラミングにおける様相オペレータの解釈は、

①節の結論部の様相オペレータ□A，◇A、

・世界wにおいて節の結論部に□Aが定義されているとき、その節の条件部が成り立つならば、wに相対的に（wからみて）可能であるすべての世界で、Aは真となる。より具体的には、世界wに相対的に可能なすべての世界でAを呼び出すことができると手続き的に解釈する。

・世界wにおいて節の結論部に◇Aが定義されているとき、その節の条件部が成り立つならば、wに相対的に可能な世界で、Aが真となる世界が存在する。このAが真となる幾つか存在する世界をどのように決定するかの問題がプログラミング言語上生じるが、ここでは以下の2通りの方法で解釈・処理する。

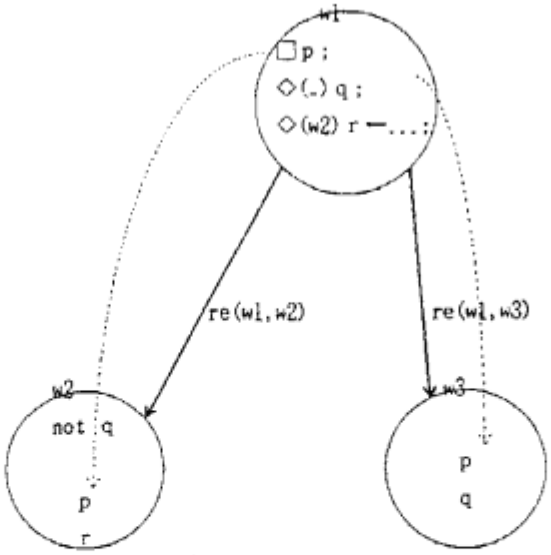
a）プログラミング言語上のシンタックス◇(I)Aにより、変数Iに指定される世界で、Aは真となる（但し、Iにより指定される世界は、現在いる世界wから到達可能な世界でなければならない）。

b）◇(I)Aで、Iに指定がない時は、wから到達可能で矛盾が起こらない（すなわち、その到達可能な世界でnot Aが成り立っていない）すべての世界でAが真となる。

このように◇(I)Aは、世界wに相対的に可能な世界のうちで、矛盾を起こさないかIによって指定される世界でAを呼び出すことができると手続き的に解釈される。

(例)

```
relation of
  re(w1,w2);
  re(w1,w3)
fo.
world w1 of
  □p;
  ◇(_)q;
  ◇(w2)r←s;
  s
fo.
world w2 of
  not q
fo.
world w3 of
fo.
```



世界w2ではp，not q，rが成り立ち、世界w3ではp，qが成り立つ。

②節の条件部の様相オペレータ□A，◇A、

・世界wにおいて節の条件部中に□Aが定義されているとき、□Aがwで真と定義されているか、又はwに相対的に可能であるすべての世界でAが真となるならば、□Aは成り立つ。Aが変数を含む場合、様相述語論理における限量の問題が生じるが、ここでは議論を単純にするため、すべての世界の個体領域は（プログラムの）エルブラン空間1つに固定し、個体項の指示対象はすべての世界を通じて変化しないとする。

・世界wにおいて節の条件部中に◇Aが定義されているとき、◇Aがwで真と定義されているか、又はwに相対的に可能な世界でAが真となる世界が存在するならば、◇Aは成り立つ。やはりAが変数を含む場合は、関係(relation)定義中の順序に従って、見つかった順に値が代入されるか、シンタックス◇(I)Aでの1による指定世界における値が代入されるとする。

(例)

```
relation of
  re(w1,w2);
  re(w1,w3)
fo.
world w1 of
  p(X)←□q(X);
  r(X)←◇(_)s(X);
  t(X)←◇(w3)s(X)
fo.
world w2 of
  q(a);
  q(b);
  s(a)
fo.
world w3 of
  q(b);
  q(c);
  s(b)
fo.
```

世界w1では、p(b), r(a), r(b), t(b) が成り立つ。

〔プログラムの実行〕

・プログラムの実行は、プログラム中で定義された1つのある世界wを現実世界とみなし、そこに質問を行うことによって始まる。

以上の様相論理プログラムの定義よりその特徴は、様相論理の多世界モデル論的解釈に基づく「世界」という概念を導入することによって、論理プログラミングの構造化・モジュール化を行い、また世界とは別にその世界間の相対的可能性の関係をrelation文で定義し、様相オペレータの付いた述語により世界間の通信が行われるようになっていると述べられる。さらにこのrelation文中に、世界間の関係の関係をホーン節で定義できることが大きな特徴である。このことにより、様相オペレータの手続き的解釈と関係 re のさまざまな定義により、様相論理におけるT, S4, S5系等が容易に実現でき、さらに各々異なった様相系のインタープリタ（公理と推論規則）をその都度システムで用意する必要がない（すなわち、言語でそれを記述できる）。またオブジェクト指向やモジュール化などのソフトウェアの分割・部品化、そして組み合わせによるその利用なども実現できる。ユーザーにとって非常に柔軟でかつ表現力に富む言語となっている。

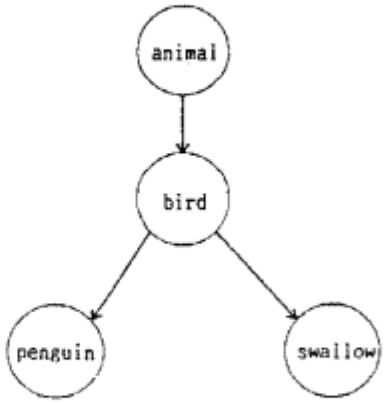
4. 応用

4.1. 知識の階層，属性の継承

知識表現において重要な考えの1つに、フレームがある。フレーム理論において中心となる考えは、フレームとスロットによる概念の構造の表現である。概念や知識の構造はフレームによって表され、その中に記述されるスロットにより推論メカニズムが表現される。さらに、スロットとその中に代入される値により、概念や知識の階層性・属性の継承のメカニズムが表現される。このフレームを様相論理プログラムにより表現するためには、様相論理プログラムにおける1つの世界が1つのフレームに対応すると考え、その世界で定義される述語が各スロットに当たるとする。そして、フレームの階層性，属性の継承は、世界間の関係reと様相オペレータ□，◇によって簡潔に記述できる。例えば、属性のis-a階層と継承は推移的なreと□オペレータにより、ディフォールトの継承は◇オペレータにより表現される。

例)

```
relation of
  re(animal,bird);
  re(bird,penguin);
  re(bird,swallow);
  re(X,Y)←re(X,Z),re(Z,Y)
fo.
world animal of
  □move;
     . . .
fo.
world bird of
  □number-of-wings(2);
  ◇(_)fly;
     . . .
fo.
world penguin of
  not fly;
     . . .
fo.
```



```
world swallow of
    . . .
fo.
```

世界penguin に←move,number-of-wings(X) を質問すると成功し、答え X=2 が返ってくるが、質問←fly は失敗し、not fly が成り立つ。一方、世界swallow ではnumber-of-wings(2), fly 等が成り立つ。

4.2. 知識の共有

人工知能におけるmulti-agent による知識の共有やあるいは分散型データベースへの応用に、様相論理プログラムが適用される。これは各世界wに定義される論理式に対して、世界wがその知識を所有していると解釈する。そして多世界モデルにより、複数のagent が自分の持つ知識を互いに共有し通信することを容易に記述できる。

(例) ([5])

2つのagent a1,a2 が持っている知識が、次の様相論理プログラムにより与えられる。

①
```
relation of
  re(a1,a2);
  re(a2,a1)
fo.
world a1 of
  p;
  q←p
fo.
world a2 of
  p←◇(a1)p;
  s←p
fo.
```

②
```
relation of
  re(a1,a2);
  re(a2,a1);
  re(a,a1); re(a,a2);
fo.
world a1 of
  p;
  q←p
fo.
world a2 of
  s←p
fo.
world a of
  ◇(a2)p←◇(a1)p
fo.
```

①の定義では、世界a2で定義される節 p←◇(a1)p がagent a1とagent a2の間でなされる通信を表現している。これは、agent a2がその通信の手段を知っている。agent(世界) a2に質問←s をすると成功する。一方、②の定義では、 a というa1,a2 を制御する (神様のような) agent がいて、これがa1とa2の通信手段を知っている。

4.3. オブジェクト指向プログラミング

オブジェクト指向プログラミングでは、オブジェクトと呼ばれる"もの"とその間で行われるメッセージ交換を中心にしてプログラミングが行われる。そのオブジェクトは、記述する対象を表すデータ構造とそれを操作する手続きから成る。

様相論理プログラムでオブジェクト指向プログラムを表現しようするとき、様相論理プログラムにおける1つの世界が1つのオブジェクトに相当する。節の結論部の様相オペレータ□, ◇の付いたアトムが、その世界へのメッセージ (手続き呼び出し) に対するインターフェイスになる。したがって、メッセージ交換は関係reのある世界間での様相オペレータによる作用となる。一方、様相オペレータの付かない述語定義は、メッセージに対する処理方法の内部での表現となり、外から (他の世界から) は見えないこととなる。これは情報隠蔽の効果をもたらし、プログラミングにおけるモジュール化を実現する。

オブジェクト指向プログラミングにおいてもう1つの重要な考えに、クラス・インスタンスの概念がある。クラスはいくつかのオブジェクトに共通した性質をもつものの集まりとして捉えられ、インスタンスはそのクラスに属する実体となる。インスタンスは自分がどのクラスに属しているかだけを知っており、あとは自分に固有のデータを内部に抱えているだけである。クラス間のis-a階層は、4.1.と同じ方法で実現できる

。そしてクラスとインスタンスの概念は、メッセージに対する手続きだけを記述するクラスとしての世界とその下のインスタンスとしての世界を用意することにより達成される。

(例)

```
world sample of
  □p←...;              手続き呼び出し，外界とのインターフェイス
  ◇(_)q←...;
      r←...;           内部処理，（情報隠蔽）
      s←...
fo.
```

5．　他の研究との比較

5.1. MOLOG

MOLOG［5］では、様相論理との融合によるPROLOGの拡張を行い、その構文に様相オペレータを導入している。推論規則は導出原理によるスタイルを取り、したがって様相オペレータの解釈は公理論的解釈による。とくにその論理の導出方法として、構文解析時における翻訳(COMPILATION) という新しい方法を導入し、記述されたプログラムをホーン節に近い文に一旦翻訳してから実行する。様相論理を取り込んだプログラミングに対する様々な議論が行われているが、具体的な導出方法はknowという１つの様相オペレータを持ったＳ５の特定の様相論理系に対してのみ与えられている。これに対して、今回提案した様相論理プログラミングでは、世界という概念とその間の関係の記述を言語に導入することによって、様相オペレータのモデル論的解釈を行っている。またこれにより、特定の様相系だけというのではなく様々な様相系の解釈・実行が行える。ここが大きく異なる点であり、今回の提案の特徴を表していると言える。

5.2. Prolog/KR

Prolog/KR［2］は、PROLOGを基本とし、それにLISPの持つ制御構造や、多重世界機能を追加し、拡張した、知識表現に適したプログラミング言語である。そのプログラムはいくつかの互いに独立な世界より構成される。各々の世界で定義された述語は外側からは見えず、それらを起動するにはwithというプリミティブを用いて、その世界に入り呼びだす。また世界は何重にもネストして用いることができ、その場合内側の世界からは外側の世界の定義が全部見える。また世界間のネスティングが各世界の定義とは独立に定義時に決められ、それは様相論理プログラムでのrelation文に似ている。しかし多重という言葉が示す通り、世界間の関係はネスティングによるものだけであり、内側の世界からは外側の世界が（原則として）すべて見えてしまう。これはモデル論的解釈によれば、Ｓ４に相当すると考えられる。これに対して、様相論理プログラムにおける世界は、Prolog/KR の世界とほぼ同じ概念であるが、relation文によってＳ４ばかりでなくより柔軟な世界間の関係が記述でき、また世界間で見えるのは様相オペレータの付いた述語だけであり、他は全て外から隠されている。ただProlog/KR では、様々な制御用述語が用意してあり、プログラミング言語として使いがってが良く、また知識表現におけるプログラミング経験が豊富であり、今後も大いに参考になると考えられる。

6．　まとめ（今後の課題・拡張）

①様相オペレータ□，◇による限量の範囲の拡大。すなわち、節全体を様相オペレータで限定する記述も許す。

(例)

□（p←q，r，s），◇（p←q，□（r，s））等。

②ユーザーが必要とする種類だけの様相オペレータと各様相ごとの関係reを定義できるように拡張する。これにより、Ｓ４やＳ５など様々な異なる種類の様相概念を同時に扱うプログラミングの記述が可能となる。またいくつかの様相オペレータの１つとして、時間や状態の変化を扱える様相オペレータを用意する。

これにより、assertなどのより形式的な扱いが可能となる（［6］）。さらに何種類かの様相オペレータとその各様相の関係reを用意することにより、意味ネットワークのような知識表現も可能となる。

（例）

様相オペレータ　know, assume, □, ◆　等

その関係　　re-know(w1,w2), re-assume(w1,w2), re(w1,w2), re'(w1,w2)　等。

③仮想的に無数の（可能）世界を作りだすメカニズム（機構）を用意する。例えば、時間を扱う様相概念を多世界モデルにより形式的に表現しようとするとき、それは過去から未来にわたる無限の世界を用意する必要がある。したがって、再帰的な定義や変数に代入される値の変化により無数の世界があたかも存在するように仮想的に見せるメカニズムが必要とされる。これにより時間や状態の変化なども形式的に扱えるようになる。

④並列論理プログラミングへの応用

可能多世界の概念を持つ様相論理プログラミングは、ＯＲ並列の記述に適している。ＯＲ並列性を持つ述語に様相オペレータ◇を付け、その世界から相対的に可能ないくつかの世界にその述語に対するそれぞれ異なった可能な定義をする。

⑤非単調論理，知識の論理への応用

⑥否定not の取り扱い

失敗による否定，論理的否定，それらの違いの意味的考察など。

⑦プログラミング言語としての様相論理プログラミングの形式的意味論の確立。

尚、本研究は、第５世代計算機プロジェクトの一環として、ICOTの委託で行ったものである。

（参考文献）

［1］神野他：論理学ーモデル理論と歴史的背景ー，ミネルヴァ書房，1976.

［2］中島秀之：知識表現とProlog/KR，産業図書，1985.

［3］Bowen,K.A.：Meta-Level Programming and Knowledge Representation, New Generation Computing, 3(1985), 359-383.

［4］Chikayama,T.：ESP Reference Manual, ICOT Technical Report TR-044, 1984.

［5］Farinas del Cerro,L.：MOLOG: A System That Extends PROLOG with Modal Logic, New Generation Computing, 4(1986), 35-50.

［6］Warren,D.S.：DATABASE UPDATES IN PURE PROLOG, Proc. of FGCS 1984, 244-253.