知識工学と人工知能 34－4
（１９８４．５．８）

Ｆｕｚｚｙ論理を組み込んだＰｒｏｌｏｇ－ＥＬＦ

金井　直樹，石塚　満
（東京大学　生産技術研究所）

１．はじめに

プログラミング言語ＰＲＯＬＯＧは，一階述語論理を基にした言語であり，論理言語の特徴をいかし，知的システム作成への応用が行われている．しかし，基本的にはＰＲＯＬＯＧは二値論理を扱うため，多値論理を応用した問題に使用するためには，インプリメント時に様々な工夫が要求される．今後，不確実さ／あいまいさを含むような情報や知識を扱うような知的システムを作成するためには，その基礎となるプログラミング言語を持つことが望ましい［１］．

ファジイ論理は，二値論理と異なり，真理値として０以上１以下の実数をとり，あいまいさを取り扱うことができる．我々は，ファジイ述語論理を容易に扱えるように拡張したＰＲＯＬＯＧ－ＥＬＦ（ＥＸＴＥＮＤＥＤ　ＰＲＯＬＯＧ　ＴＯ　ＦＵＺＺＹ　ＬＯＧＩＣ）を開発し，有用な機能を追加した［２］．今までにもファジイ情報を扱う言語が作成されているが［３，４］，ＥＬＦは，知識型システムの基礎として高い評価を得つつあるＰＲＯＬＯＧの機能を含み，その拡張として作成されている．本報告では，ＥＬＦの概要と，その応用例について述べる．

２．ファジイ論理

ファジイ論理は，一般にファジイ集合を対象にすることができるが［５，６］，ここでは通常の記号で考える．ファジイ論理では，ある論理式の真理値Ｔは，０≦Ｔ≦１であり，その論理式の正しさの程度を表わす．ファジイ論理での真理値の取り扱いは，次のようになる．論理式をＳとした時，Ｓの真理値をＴ（Ｓ）で表わすことにする．

（2.1）ＩＦ　Ｓ＝Ａ　かつ　Ａが素論理式：Ｔ（Ｓ）＝Ｔ（Ａ）
（2.2）ＩＦ　Ｓ＝～Ｒ　　：Ｔ（Ｓ）＝１－Ｔ（Ｒ）
（2.3）ＩＦ　Ｓ＝Ｓ１　ＡＮＤ　Ｓ２　　：Ｔ（Ｓ）＝ＭＩＮ（Ｔ（Ｓ１），Ｔ（Ｓ２））
（2.4）ＩＦ　Ｓ＝Ｓ１　ＯＲ　Ｓ２　　：Ｔ（Ｓ）＝ＭＡＸ（Ｔ（Ｓ１），Ｔ（Ｓ２））

たとえば，

Ｔ（Ｐ）＝0.1，　Ｔ（Ｑ）＝0.7，　Ｔ（Ｒ）＝0.6

である時，Ｓが，

Ｓ＝（Ｐ　ＯＲ　Ｑ）　ＡＮＤ　（～Ｒ）

で表わされるとすると，Ｓの真理値Ｔ（Ｓ）は，

Ｔ（Ｓ）＝ＭＩＮ（ＭＡＸ（Ｔ（Ｐ），Ｔ（Ｑ）），１－Ｔ（Ｒ））
＝ＭＩＮ（ＭＡＸ（0.1，0.7），１－0.6）
＝ＭＩＮ（0.7，0.4）＝0.4

のように求められる．ファジイ論理では，Ｔ（Ｓ）≧0.5の時ファジイ恒真，Ｔ（Ｓ）≦0.5の時ファジイ矛盾と呼ぶ．

3．ファジイ論理とＰＲＯＬＯＧ

二値論理において，論理式

Ｐ，　Ｐ→Ｑ

が真の時，この結果，Ｑが真であることがわかる．これがｍｏｄｕｓ　ｐｏｎｅｎｓである．ファジイ論理では，真理値は０以上１以下の実数で表わされる．上記の二つの論理式において，

Ｔ（Ｐ）＝ａ，　Ｔ（Ｐ→Ｑ）＝ｂ

であるとする．ファジイ論理のｍｏｄｕｓ　ｐｏｎｅｎｓでも同様にして，

（Ｐ　ＡＮＤ　（Ｐ→Ｑ））→Ｑ

としてＱがえられる．この時，Ｑの真理値は，（2.3）より，

Ｔ（Ｑ）＝Ｔ（Ｐ　ＡＮＤ　（Ｐ→Ｑ））

＝ＭＩＮ（Ｔ（Ｐ），Ｔ（Ｐ→Ｑ））

＝ＭＩＮ（ａ，ｂ）

として求められる．

一方，ＰＲＯＬＯＧは，導出原理を基にしたプログラミング言語である．その実行過程は，ｍｏｄｕｓ　ｔｏｌｌｅｎｓとしてとらえることができる．プログラムはリテラルＰ，Ｑがあった時，

Ｐ　(1)

Ｐ→Ｑ　(2)

の形で与えられる．(1)は，Ｐが真であることを意味する．(1)，(2)の形のプログラムをアサートして公理の集合をつくる．あるリテラルの真偽を調べる時には，そのリテラルの否定を公理の集合に加え，矛盾を導き出すことにより，そのリテラルが真であることを求める．たとえば，(1)，(2)がある時，Ｑの真偽を調べるには，

～Ｑ　(3)

を公理の集合に加える．(2)と(3)を用いると，

（（～Ｑ）ＡＮＤ（Ｐ→Ｑ））→～Ｐ

より，

～Ｐ　(4)

が導かれる．(1)と(4)より，排中律

Ｐ　ＡＮＤ　（～Ｐ）　(5)

が導かれると，矛盾が生じたことになり，(3)が否定されＱが真であることがわかる．

また，この考え方と逆に，あるリテラルの真偽を問うと，そのリテラルを導く論理式の集合を探し出すシステムとして，ＰＲＯＬＯＧをとらえることができる．たとえば，Ｑの真偽を問うと，(1)と(2)という論理式が探し出され，Ｑが真であることが示されるという考え方である．

このように考えると，ファジイ論理を容易に，ＰＲＯＬＯＧで扱うことができる．それぞれの論理式に０以上１以下の真理値が与えられている時，あるリテラルの真理値を知りたい場合，そのリテラルを導く論理式の集合を求め，個々の論理式の真理値に対して，２で述べた真理値の取り扱いを適用することにより，そのリテラルの真理値を求めることができる．

具体的には，たとえば，

P，　　T（P）＝a　　(6)

P→Q，　　T（P→Q）＝b　　(7)

がある時，Qは(6)，(7)より導くことができるので，Qの真理値は，

（P　AND　（P→Q））→Q

T（Q）＝MIN（a，b）

として求められる．このような通常のファジイ論理に，変数を持つ論理式を含めるようにすると，導出原理などの機能を持つファジイ述語論理を定義できる［7，8］．

ただし，定義節中に変数が含まれる場合，ユニフィケーションの際にその定義節の真理値をどう扱うかという問題がある．定義節の真理値が，ユニフィケーションによって，変化しないと仮定すれば，問題は生じない．また，次のような問題もある．

（3.1）ORの問題：

2の定義からわかるように，ファジイ論理のORでは，いくつかの論理式があった時，そのうち真理値が最大のものの中から一つを選択する．PROLOGにファジイ論理を組み込んだ場合，入力したゴール節を導く論理式の集合が複数あった時には，そのゴール節の真理値が最大になるような集合を選択する必要がある．このためには，PROLOGの実行過程において全数探索を行うことが要求される．ただし，次のような問題もある．たとえば，

R（＊X）←P（＊X）AND　Q（＊X）……真理値は0.9

P（a）……真理値は0.9

P（b）……真理値は0.7

Q（a）……真理値は0.5

Q（b）……真理値は0.8

があったとすると，

T（R（a））＝MIN（0.9，MIN（0.9，0.5））＝0.5

T（R（b））＝MIN（0.9，MIN（0.7，0.8））＝0.7

となる．R（＊）に関して，真理値最大のものをとると，P（＊）に関しては真理値最大のものをとれないことになる．つまり，ユーザは，トップレベルでは最大の真理値を選択することが可能だが，サブゴールでは最大の真理値を選択することができず，トップレベルとサブゴールの取り扱いに関し一貫性がとれない．

（3.2）NOTの問題：

2の定義より，Pの真理値が0.8ならば，～Pの真理値は0.2になる．PROLOGにおいては，変数が他の項とユニファイされることがある．従来のPROLOGでは，NOTで実行されるゴール節中の変数は，その実行によって他の項とユニファイされることはなかった．しかし，ファジイ論理を扱うと，NOTの実行中に行われたユニフィケーションの結果が残ることがある．たとえば，

P（a）……真理値は0.7

の時，

NOT（P（＊X））

を実行すると，

ＮＯＴ（Ｐ（ａ））……真理値は0.3

となり，＊Ｘにａがユニファイされて結果が得られる．また，Ｐ（＊Ｘ）を導く定義節の集合が複数あった場合には，ＯＲの関係にあることを考慮して真理値が最大になる集合のうち一つを選択して，そのＮＯＴをとる必要がある．

４．ＥＬＦ

ＥＬＦは，ファジイ述語論理を扱うように拡張したＰＲＯＬＯＧであり，そのために要求される機能をつけ加えた．ＥＬＦは，ＶＡＸ１１／７８０上で，ＢＥＲＫＥＬＥＹ－ＰＡＳＣＡＬを用いてインプリメントした［＊］．

ＥＬＦでは，アサートされた節にそれぞれ真理値を属性として持たせることにする．この時，ユニフィケーションに対して，その節の真理値は変わらないものとする．また，項の状態によって動的に真理値を設定する方法も用意した．真理値の計算は，３で述べたように，結論を導き出す節の集合を探し，それぞれの節の真理値に対して，２で述べた操作を加えて行う．

ＥＬＦでは，節は，

定義節　：　＋Ｐ－Ｑ－……－Ｒ．　　又は，　　＋Ｐ．

ゴール節：　－Ｑ－……－Ｒ．

で表わされる．真理値の設定は，

0.6：＋Ｐ－Ｑ．　　又は，　　－ＡＳＳＥＲＴ（0.6：＋Ｐ－Ｑ．）．

のように，アサートする節の前に真理値を与えることにより行う．省略した場合は，真理値として１が設定される．従って，すべての節の真理値が省略されている場合は，通常のＰＲＯＬＯＧと同一の動作をする．真理値として０以下，あるいは，１より大きい値が与えられるとエラーになる．ゴール節の実行結果は，真理値が０より大きい時，

0.7：－Ｐ．

の形式で出力される．

次に，ＥＬＦに加えた機能と組み込み述語について述べる．

（4.1）ＵＳＥＶＡＬＵＥ：

真理値を動的に設定するための述語．引数で与えられた値を，その節全体の真理値として扱う機能を持つ．ファジイ集合ＯＬＤの定義の例を以下に示す．

```
+old(*age)-gt(*age 80)-/-usevalue(1).
+old(*age)-gt(*age 60)-/-minus(*age 60 *x)
      -times(*x 0.01 *y)-plus(*y 0.8 *z)
      -usevalue(*z).
+old(*age)-gt(*age 40)-/-minus(*age 40 *x)
      -times(*x 0.035 *y)-plus(*y 0.1 *z)
      -usevalue(*z).
+old(*age)-times(*age 0.0025 *x)
      -usevalue(*x).
```

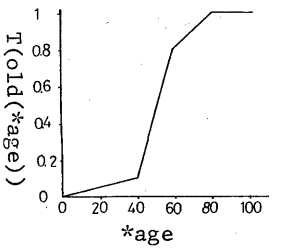

図１．ＵＳＥＶＡＬＵＥを用いたＯＬＤの定義の例

（4.2）ＶＡＬＵＥ：

ある節の真理値を求める機能を持ち，その真理値を第２引数に返す．非決定的な述語である．たとえば，

0.7：＋Ｐ（ａ）．　　0.5：＋Ｐ（ｂ）．　　(8)

がある時，

－ＶＡＬＵＥ（－Ｐ（＊Ｘ）．，＊ＶＡＬ）．

を実行すると，初めは，＊Ｘとａ，＊ＶＡＬと0.7がユニファイされる．バックトラックが発生すると，＊Ｘとｂ，＊ＶＡＬと0.5がユニファイされる．次のバックトラック時には，＊ＶＡＬと０がユニファイされる．その後，さらにバックトラックが発生すると，ＶＡＬＵＥの実行が失敗する．具体例を以下に示す．

```
>+very(*cl)-value(*cl *x)-sqr(*x *y)
          -usevalue(*y).
success
>-old(61).
 0.8100:-old(61).
>-very(-old(61).).
 0.6561:-very(-old(61).).
```

図２．ＶＡＬＵＥを用いた例

（4.3）ＣＨＶＡＬＵＥ：

一度，アサートした節の真理値を再定義する機能を持つ．

（4.4）ＴＨＲＥＳＨ：

引数で与えられた値をスレッシュホールドに設定する．実行中に，真理値がそれ以下になると偽と見なし，バックトラックが発生する．

（4.5）ＭＡＸＣＬ，ＤＭＡＸＣＬ：

一つ以上のゴール節を引数にとる．それらのゴール節を実行し，真理値が最大になる結果を返す．たとえば，(8)がある時，

－ＭＡＸＣＬ（－Ｐ（＊Ｘ）．）．

を実行すると，＊Ｘとａがユニファイされ，このＭＡＸＣＬの真理値として，0.7が得られる．ＭＡＸＣＬは非決定的な述語であり，ＤＭＡＸＣＬは，決定的な述語である．つまり，ゴール節の実行によって，真理値が最大の結果が複数得られた時，その後，バックトラックが発生した場合には，ＭＡＸＣＬでは他の結果を用いて実行を続けるが，ＤＭＡＸＣＬでは，失敗することになる．これらの述語を用いることにより，（3.1）で述べた問題に対処でき，真理値最大のものを用いるファジイ論理のＯＲを実現できる．

（4.6）ＮＯＴ：

ＥＬＦのＮＯＴでは，引数のゴール節を実行し，ＤＭＡＸＣＬのように真理値最大の結果のうちの一つを選び，１からその真理値を引いた値をそのＮＯＴの真理値とする．ゴール節の真理値が０か１の時は，通常のＰＲＯＬＯＧと同じ動作をするが，それ以外の時には，ゴール節実行中に行われたユニフィケーションの結果が残る．

（4.7）３つのモードが設定できる．

（4.7.1）ＮＯ－ＱＵＥＲＹモード：

一番始めに得られた結論を表示し実行を中止する．

（4.7.2）ＱＵＥＲＹモード：

すべての結論を表示する．重複解も許す．

（4.7.3）ＢＥＳＴ（Ｎ）モード：

得られた結論のうち，真理値が大きい順にＮ個表示する．

全数探索の範囲を小さくするため，ＢＥＳＴ（Ｎ）モードでは，ユーザが指定したスレッシュホールドとは別の，実行中に動的に変わるスレッシュホールドを使用する．ＢＥＳＴ（Ｎ）モードで実行中に，すでにＮ個以上の結果が得られている場合，その結果の真理値の最小値を考慮すると，それ以降，真理値がそれ以下の結果は必要がない．このため，その最小値を動的に変化するもう一つのスレッシュホールドとして使用する．

5．ＥＬＦの実行例

以下にＥＬＦの実行例を示す．左がプログラム，右が実行例である．これは，別荘を持っている人を推論する例である．

```
; Have a second house ;

+have_a_second_house(Kato).
0.9:+have_a_second_house(*who)
        -rich(*who).
0.7:+have_a_second_house(*who)
        -president(*who).

; Rich ;

+rich(*who)
    -income(*who,*yen)-rich2(*yen).
+rich2(*yen)-gt(*yen,20000000)-/.
+rich2(*yen)-gt(*yen,10000000)-/
    -minus(*yen,1e7,*x)-rdiv(*x,1e8,*y)
    -times(3,*y,*z)-plus(0.7,*z,*w)
    -times(3,*y,*z)-plus(0.7,*z,*w)
    -usevalue(*w).
+rich2(*yen)-rdiv(*yen,1e8,*x)
    -times(7,*x,*y)-usevalue(*y).

; Income ;

+income(Yamada,15000000).
+income(Sato,30000000).
+income(Nomura,8000000).
+income(Kobayashi,40000000).

; President ;

+president(Suzuki).
+president(Kobayashi).
```

```
fuzzy-prolog ELF version 2.1    84/3/27
>-load(house_).
>-have_a_second_house(*).
 1.0000:-have_a_second_house(Kato).
>-best(10).
 1.0000:-best(10).
>-have_a_second_house(*).
 1.0000:-have_a_second_house(Kato).
 0.9000:-have_a_second_house(Sato).
 0.9000:-have_a_second_house(Kobayashi).
 0.8500:-have_a_second_house(Yamada).
 0.7000:-have_a_second_house(Suzuki).
 0.5600:-have_a_second_house(Nomura).
```

ＥＬＦは，関係データベースの検索にも容易に応用できる［９］．以下にその例を示す．上がプログラム，下が実行例である．？は，数学が７０点以上で物理の成績が良い人を，？２は，数学が７０点以上で物理の成績が非常に良い人を探すための述語である．

```
; Mathematics ;

+mathematics(TANAKA 74).
+mathematics(SUZUKI 71).
+mathematics(KATO 65).
+mathematics(SATO 86).
+mathematics(YAMADA 79).
+mathematics(KONDO 75).
+mathematics(YOSHIDA 63).
+mathematics(MITAKA 66).
+mathematics(YOTSUYA 65).
+mathematics(KANDA 45).
+mathematics(HONGO 30).
+mathematics(SHIBUYA 75).
+mathematics(MEGURO 69).
+mathematics(OTSUKA 74).
+mathematics(UENO 60).
+mathematics(NEZU 48).
+mathematics(TOYOTA 72).
+mathematics(NAKANO 57).
```

```
; Physics ;

+physics(TANAKA 75).
+physics(SUZUKI 85).
+physics(KATO 61).
+physics(SATO 62).
+physics(YAMADA 59).
+physics(KONDO 68).
+physics(YOSHIDA 84).
+physics(MITAKA 66).
+physics(YOTSUYA 68).
+physics(KANDA 62).
+physics(HONGO 45).
+physics(SHIBUYA 72).
+physics(MEGURO 80).
+physics(OTSUKA 63).
+physics(UENO 75).
+physics(NEZU 54).
+physics(TOYOTA 68).
+physics(NAKANO 82).
```

```
; Good ;

+good(*p)-gt(*p 89)-/-usevalue(1).
+good(*p)-gt(*p 39)-/-minus(*p 40 *val1)
        -times(*val1 0.016 *val2)-plus(*val2 0.2 *val3)-usevalue(*val3).
+good(*p)-times(*p 0.005 *val1)-usevalue(*val1).
```

```
; Very ;

+very(*cl)-value(*cl *val)-sqr(*val *val2)-usevalue(*val2).
```

```
; Test ;

+?(*who)-mathematics(*who *p1)-physics(*who *p2)
        -gt(*p1 70)-good(*p2).
+?2(*who)-mathematics(*who *p1)-physics(*who *p2)
        -gt(*p1 70)-very(-good(*p2).).
```

```
fuzzy-prolog ELF version 2.1    84/3/27
>-load(demo_).
>-?(*).
 0.7600:-?(TANAKA).
>-best(5).
 1.0000:-best(5).
>-?(*).
 0.9200:-?(SUZUKI).
 0.7600:-?(TANAKA).
 0.7120:-?(SHIBUYA).
 0.6480:-?(KONDO).
 0.6480:-?(TOYOTA).
```

```
>-?2(*).
 0.8464:-?2(SUZUKI).
 0.5776:-?2(TANAKA).
 0.5069:-?2(SHIBUYA).
 0.4199:-?2(KONDO).
 0.4199:-?2(TOYOTA).
>-thresh(0.65).
 1.0000:-thresh(  6.500000e-01).
>-?(*).
 0.9200:-?(SUZUKI).
 0.7600:-?(TANAKA).
 0.7120:-?(SHIBUYA).
>-?2(*).
 0.8464:-?2(SUZUKI).
```

6．おわりに

ＥＬＦは，ファジイ述語論理を扱うＰＲＯＬＯＧである．ＥＬＦを用いることにより，不確実／あいまいな情報，知識を含む関係データベース［９］や，エクスパートシステム［１０］などの知識型システムの開発がより容易になることが期待される．ただし，現バージョンのＥＬＦでは，多くの場合，全数探索をしているため，適切なスレッシュホールドの設定など，実行効率の向上を図る必要があろう．

［＊］基本的なＰＲＯＬＯＧインタープリタは，中島秀之氏（電総研）が製作した．

謝辞

このＰＲＯＬＯＧインタープリタの利用を快く許可して下さった中島秀之氏に感謝します．又，日頃，熱心に討議していただく安田教授をはじめ安田研，石塚研の諸氏に感謝します．

参考文献

［１］石塚　満，〝不確かな知識の取り扱い〟，計測と制御，ＶＯＬ２２，ＮＯ９，７７４－７７９，ＳＥＰ．１９８３．

［２］金井，石塚，〝Ｆｕｚｚｙ論理を組み込んだＰｒｏｌｏｇの実装〟，情報処理学会第２８回全国大会，７Ｈ－７，１９８４．

［３］Ｌ．Ａ．ＺＡＤＥＨ，〝ＰＲＵＦ－Ａ Ｍｅａｎｉｎｇ Ｒｅｐｒｅｓｅｎｔａｔｉｏｎ Ｌａｎｇｕａｇｅ ｆｏｒ Ｎａｔｕｒａｌ Ｌａｎｇｕａｇｅ〟，ＩＮＴ．Ｊ．ＭＡＮ－ＭＡＣＨＩＮＥ ＳＴＵＤＩＥＳ，１０，３９５－４６０，１９７８．

［４］Ｍ．ＵＭＡＮＯ，Ｍ．ＭＩＺＵＭＯＴＯ，Ｋ．ＴＡＮＡＫＡ，〝ＦＳＴＤＳ Ｓｙｓｔｅｍ：Ａ Ｆｕｚｚｙ－ｓｅｔ Ｍａｎｉｐｕｌａｔｉｏｎ Ｓｙｓｔｅｍ〟，ＩＮＦＯＲＭＡＴＩＯＮ ＳＣＩＥＮＣＥＳ，１４，１１５－１５９，１９７８．

［５］Ｌ．Ａ．ＺＡＤＥＨ，〝Ｆｕｚｚｙ Ｌｏｇｉｃ ａｎｄ Ａｐｐｒｏｘｉｍａｔｅ Ｒｅａｓｏｎｉｎｇ〟，ＳＹＮＴＨＥＳＥ，３０，４０７－４２８，１９７８．

［６］Ｄ．ＤＵＢＯＩＳ，Ｈ．ＰＲＡＤＥ，〝Ｆｕｚｚｙ Ｓｅｔ ａｎｄ Ｓｙｓｔｅｍｓ：Ｔｈｅｏｒｙ ａｎｄ Ａｐｐｌｉｃａｔｉｏｎｓ〟，ＡＣＡＤＥＭＩＣ ＰＲＥＳＳ，１９８０．

［７］Ｒ．Ｃ．Ｔ．ＬＥＥ，〝Ｆｕｚｚｙ Ｌｏｇｉｃ ａｎｄ Ｒｅｓｏｌｕｔｉｏｎ Ｐｒｉｎｃｉｐｌｅ〟，Ｊ．ＡＣＭ １９，１０９－１１９，ＪＵＮ．１９７２．

［８］向殿，増沢，〝Ｆｕｚｚｙ論理における導出形の性質について〟，信学論（Ｄ），Ｊ６６－Ｄ，７９６－８０３，１９８３．

［９］Ｃ．Ｌ．ＣＨＡＮＧ，〝Ｄｅｃｉｓｉｏｎ Ｓｕｐｐｏｒｔ ｉｎ ａｎ Ｉｍｐｅｒｆｅｃｔ Ｗｏｒｌｄ〟，ＡＵＴＯＭＡＴＩＮＧ ＩＮＴＥＬＬ．ＢＥＨＡＶＩＯＲＥ：ＡＰＰＬＩＣＡＴＩＯＮＳ ＡＮＤ ＦＲＯＮＴＥＥＲＳ．ＡＴ ＧＥＩＴＨＥＲＳＢＵＲＧ，１９８３．

［１０］石塚，〝建築物被害査定のエクスパートシステム〟情報処理学会論文誌，２４，３５７－３６３，１９８３．