

Prolog-ELF: ファジィ論理を組み込んだ Prolog†

金井直樹** 石塚満††

不確実性やファジィ性を伴う知識を扱う知識型システムの基礎言語として、ファジィ論理と幾つかの有用な機能を Prolog に組み入れた Prolog-ELF と名付けた言語を開発した。Prolog-ELF は Prolog のすべての好ましい基本的な性質を受け継いでいる。節形式のアサーションは 1.0 から 0.5 の間 (オプションとして 1.0 から 0.0 まで可) の真理値付きでできるのに加え、ファジィ集合もある範囲内で非常に容易に扱うことができる。応用としてファジィ論理データベースを例示している。

1. ま え が き

知識型システムの基礎言語として、Prolog はパターンマッチング (ユニフィケーション)、自動バックトラッキング、関係データベース機能など、多くの好ましい性質を有している。最近では多くの知識型システムが Prolog によって構築されるようになってきている。

1 階述語論理 (さらに詳しく言うと、1 階述語論理の部分集合であるホーン論理) を基礎とする Prolog は、基本的に 2 値論理、すなわち真 (1) が偽 (0) かをとる問題を対象にしているが、現実の問題では知識はしばしば不確実性を伴うことがある。このような不確実さ¹⁾ や不完全な知識の活用も図るのが知識工学の一つの特徴であり、不確実な知識は Mycin²⁾, Prospector³⁾, Casnet⁴⁾, Speril^{5),6)} など多くのエキスパートシステムで重要な役割を果たしている。Prolog で不確実性を扱おうとすると、特別なプログラミング技術を必要とし⁷⁾、これは大規模システムの構築には煩わしく、不確実性を扱える能力をもつ知識型システム用基礎言語が必要と考えられる。

ファジィ論理では不確実性を伴う事実やルールは、1 から 0 の間の真理値付きで表される⁸⁾。もちろん、Mycin の CF²⁾、主観的ベイズの方法⁹⁾、Dempster-Shafer 理論^{6),9)} のように、他の不確実性を扱う方法も存在する¹⁾。しかしファジィ論理はこれらの方法とは異なり、論理的な側面をもっており、この論理的な面は豊かな表現能力をもつ 1 階述語論理へと拡張できる。

本論文はファジィ論理を Prolog に組み込んだ

Prolog-ELF と名付けたプログラミング言語について記している¹⁰⁾。Prolog-ELF は知識型システムの基礎言語として高い評価を受けつつある Prolog のすべての好ましい基本的な性質を受け継いでいることで、これまでのファジィ集合を扱う言語^{11),12)} と異なっている。さらに、Prolog-ELF には幾つかの有用な述語が付加されており、これらの述語を用いることにより、ファジィ集合もある範囲内で容易に扱うことができる。

Prolog-ELF は Pascal によってインプリメントされており、現在のところ VAX-11 上で動作している。本論文では Prolog-ELF の設計の背景と特徴を記し、応用としてファジィ論理データベースを例示する。

2. ファジィ Prolog へ向けて

一般のファジィ論理ではファジィ集合を対象とすることができるが⁸⁾、ここではまず通常の意味の記号で表された論理式を考える。ファジィ論理では論理式の確実性は真理値 T ($1 \geq T \geq 0$) によって表される。(言語的真理値を用いる種類の異なるファジィ論理が Zadeh によって示されている¹³⁾。)

論理式 P の真理値を $T(P)$ とすると、次の真理値に関する扱いはファジィ論理の基本的なものである。(\neg は NOT を示す。)

$$P=A \text{ で } A \text{ が素論理式なら: } T(P)=T(A)$$

$$P=\neg Q \text{ なら: } T(P)=1-T(Q)$$

$$P=Q \wedge R \text{ なら: } T(P)=\min(T(Q), T(R))$$

$$P=Q \vee R \text{ なら: } T(P)=\max(T(Q), T(R)).$$

たとえば、もし

$$T(P)=0.6, T(Q)=0.9, T(R)=0.8,$$

$$S=(P \vee Q) \wedge \neg R$$

なら、 S の真理値を求めると次のようになる。

$$T(S)=\min(\max(T(P), T(Q)), 1-T(R))$$

$$=\min(\max(0.6, 0.9), 1-0.8)=0.2.$$

ファジィ論理では、交換律、結合律、分配律、De

† Prolog-ELF Incorporating Fuzzy Logic by NAOKI KANAI and MITSURU ISHIZUKA (Institute of Industrial Science, University of Tokyo).

†† 東京大学生産技術研究所

* 現在 日本 IBM(株)

Morgan の法則などの大部分の 2 値論理の法則が成立するが、排中律は成立しない。すなわち、 $T(P \wedge \neg P)$ は必ずしも 0 でなく、また $T(P \vee \neg P)$ も必ずしも 1 ではない。

Lee¹⁴⁾ と向殿¹⁵⁾ はファジィ論理における導出原理 (resolution principle) を考察している。ファジィ論理で $P \rightarrow Q$ を $(\neg P \vee Q)$ と等価であると解釈すれば (2 値論理では等価であるとしている)、広く知られている次の推論法、

Modes ponens: $P \rightarrow Q$ かつ P ならば Q

Modus tollens: $P \rightarrow Q$ かつ $\neg Q$ ならば $\neg P$

三段論法: $P \rightarrow Q$ かつ $Q \rightarrow R$ ならば $P \rightarrow R$

は導出原理の特殊ケースとすることができる。論理記号に引数を付けることを許し (論理記号は述語になる)、ユニフィケーションの機能を加えることにより、ファジィ論理のファジィ 1 階述語論理への拡張が図れる。

Lee は、もし導出に用いる親節の真理値がすべて 0.5 以上ならば、導出原理によって得られる導出節は意味のある推論結果となり、その導出節の真理値は必ず親節の真理値の最大値と最小値の間であることを示している¹⁴⁾。向殿は、たとえ親節の真理値が 0.5 以下であっても、そこから得られる導出節は曖昧さが減少しているという意味で、意味のある推論結果となるという解釈を示している¹⁵⁾。

ところで、Prolog では表現を、効率的な線形入力導出法 (linear input resolution) が完全性のある導出法として働くホーン節 (たかだか 1 個の正のリテラルをもつ節) の範囲に制限している。もしファジィ・ホーン論理で真理値 0.5 以下のホーン節を定義するならば、それはその節の否定と同様な効果をもち、ホーン節の範囲を逸脱するおそれがある。ゆえに、我々はファジィ論理を扱う Prolog を実装するに際し、基本的に論理表現は真理値 0.5 以上をもつホーン節に制限することにする。この場合、導出によって得られる結果は Lee の論文¹⁴⁾により、常に意味のある推論結果になることが保証される。

Lee は導出節の真理値は親節の真理値の最大値と最小値の間であることを保証した。この最小値を下限真理値 (lower bound of truth-value) と呼ぶことにする。本 Prolog の実現に際しては、導出節の真理値はこの下限真理値をとるものとする。すなわち、導出操作により得られる導出節の真理値は最も控えめな値に設定されるものとする。これは $t_1: P \rightarrow Q$, $t_2: P$ (ここ

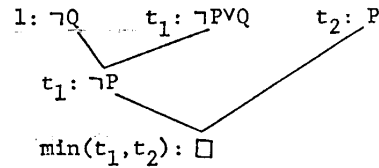


図 1 ファジィ論理における背理法の説明
Fig. 1 Explanation of the refutation mechanism in the fuzzy logic.

で t_1, t_2 は真理値) のとき、 Q は真理値 $\min(t_1, t_2)$ で導かれるとする通常のファジィ推論とも整合する。

Lee は背理法の適用については何ら示していない。Prolog としてのインプリメントは背理法によるので、我々が探るファジィ論理での考え方を示しておく。

命題論理で考え、次の公理が定義されているものとする。

$t_1: \neg P \vee Q$

$t_2: P$

これより Q を証明しようとするとき、背理法の考え方より $1: \neg Q$ (真理値 1 の $\neg Q$) を公理系に加え、空節 (矛盾) が導かれるか否か検証する。導出原理により図 1 の導出を行う。 $\neg Q$ を加えたことにより空節 (\square) が下限真理値 $\min(t_1, t_2)$ で導かれるので、逆に Q は下限真理値 $\min(t_1, t_2)$ で正しいと考える。1 階述語のときは変数に対する単一化操作が加わる。

もし変数がユニフィケーションによって具体値が代入されると、その変数を含む節の真理値が変わってしまうならば問題であるが、我々は節の真理値はユニフィケーションによって変わらないという立場をとる。すなわち、ここで考えるファジィ Prolog でも変数は全称変数で、その全範囲にわたって節を正しいものとして定義するとする。

通常の Prolog とファジィ Prolog の推論プロセスの大きな違いは、ファジィ Prolog ではしばしば OR 分岐で真理値最大のリテラルを求める必要があるために、全数探索が必要になることである。全数探索により答を求める速度が低下することになるので、不必要な部分で全数探索は実行しない、最大値を求める効率的な探索を行わせる機能などをもつことが要求される。Prolog-ELF では、これらの機能の実現を次節に記す選択可能な動作モード、動的なスレッシュホールドの導入によって図っている。

3. Prolog-ELF の仕様と機能

Prolog-ELF は前述の検討に基づいて設計、インプ

リメントされた。

Prolog-ELF の節は次の形式で記述する。

定義節: $+P-Q \dots -R$. または $+P$.

ゴール節: $-Q \dots -R$.

節の真理値は次のようにして与えることができる。

0.7: $+P-Q$.

または

$-\text{assert}(0.7: +P-Q)$.

ここで、:の前の 0.7 が真理値であり、真理値の割当てが省略されるとデフォルトの真理値 1 が設定される。もしすべての節の真理値が 1 であるならば、Prolog-ELF の動作は通常の Prolog の動作と同等となる。

節の記述を、普及している $P: -Q, R$. のような形式でなく、上記の $+P-Q-R$. のような形式とした理由は、ファジィ論理における含意 (implication) の解釈に関する問題を避けるためである。すなわち、ファジィ論理では $P \rightarrow Q$ の解釈は $\neg P \vee Q$ とは限らず、他に幾つかの解釈法が存在する^{9), 16)}。ここでの記述形式を用いることにより、我々は Prolog-ELF は節形式で定義された公理系上で動くことを明確にすることができる。もし適当ならばこれらの節形式の幾つかを含意形として解釈すればよい。

Prolog-ELF では変数は先頭に*をもつ任意の文字列で表される (単一の*も変数になり得る)。ゴール節の実行結果は、もし答の (下限) 真理値が後述するスレッシュド値より大のとき、たとえば

0.75: $-P(\text{apple})$.

のように出力される。

Prolog-ELF の実現に際しては、前節に記した要求を満たすために必要な機能、すなわち基本組込み述語の検討を行った。以下に記すのは他の Prolog にはない Prolog-ELF 特有の組込み述語であり、これらを加えることによって真理値付きの節を扱うファジィ述語論理に基づくプログラミング言語としている。(以下では答の下限真理値は単に真理値として記す。)

(1) THRESH:

引数として与えられた値をスレッシュドに設定する述語である。このスレッシュド以下の真理値をもつ節は無視される、あるいは偽と見なされ、実行中ではバックトラックが起動される。デフォルトのスレッシュド値は 0.5 である。ホーン論理の範囲を逸脱することを認識して 0.5 以下のスレッシュド値に設定することも許されている (述語

NOT に関する記述を参照)。

(2) モード設定用述語:

Prolog-ELF には三つの動作モードがあり、次の述語はこのモードを切替えるスイッチの役割を果たす。

(2-1) NO-QUERY:

システムは真理値の大きさにかかわらず、スレッシュド以上の真理値をもつ答を深さ優先探索し、最初に見つけた一つの答を出力する。これはデフォルトのモードであり、OR 分岐でも全数探索は行わない。通常の Prolog と同じ動作であり、速度の低下は (インプリメント上の差は別にして) ない。

(2-2) QUERY:

すべての可能な答を探索し、出力する。

(2-3) BEST(n):

真理値の上位 n 個の答を探索し、出力するモードである。BEST(1) に設定すると、ファジィ論理の普通の意味の答を求めることができる。このモードでは答を求める効率を向上させるために、動的なスレッシュド値が用いられ、スレッシュド値は既に得られている答の真理値の上位から n 番目の値に動的に設定される。

(3) USEVALUE:

引数として与えられた値をこの述語が使われている節の真理値に設定する。図 2 に例示されているように、この述語を用いることにより非常に容易にフ

```
+old(*age)-gt(*age,70)-/-usevalue(1).
+old(*age)-gt(*age,50)-/-minus(*age,50,*x)
  -times(*x,0.015,*y)-plus(*y,0.7,*z)-usevalue(*z).
+old(*age)-gt(*age,30)-/-minus(*age,30,*x)
  -times(*x,0.01,*y)-plus(*y,0.5,*z)-usevalue(*z).
```

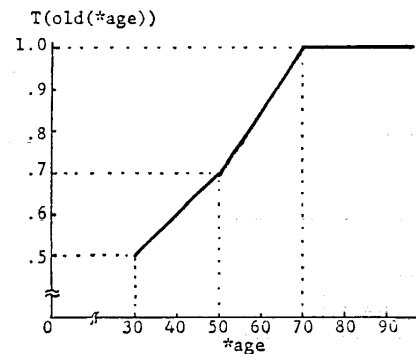


図 2 述語 USEVALUE を用いてのファジィ集合 old の定義 (/ はカット演算子を表す) と、そのグラフ表現

Fig. 2 Definition of a fuzzy set —old— using USEVALEU (/ denotes the cut operator) and its graphical representation.

ァジ集合の概念 (図2の例では old) を表現することができる。ASSERT, RETRACT が公理系を直接操作する高階の述語であるように, USEVALUE は真理値を直接操作する高階の述語である。(USEVALUE による真理値の設定を優先するので, USEVALUE を含む節に元来付されている真理値があっても無視する。) この使用は応用上便利な点が多いが, (ファジィ) 1階述語論理の枠を超えることになるので, ユーザ自身が責任をもって使用する必要がある。

(4) VALUE:

非決定的述語であり, 第1引数に与えられる節の真理値を第2引数に代入する。たとえば

0.8: +P(a).

0.6: +P(b).

が定義されていたとすると,

-value(-P(*x)., *val).

が実行されると, 最初は *x は a に, *val は 0.8 にユニファイされる。バックトラック発生時には *x は b に, *val は 0.6 にユニファイされる。図3はこの述語の使用法を示している対話の例である。

(5) CHVALUE:

CHVALUE (*x, *n, *m) の形式で, 述語名 *x の *n 番目の定義の真理値を *m に更新する。(通常エディット時に使用する。) USEVALUE と同様に真理値を直接操作する高階の述語であり, ユーザ自身が責任をもって使用する。

(6) MAXCL, DMAXCL:

ファジィ論理では最大の真理値をもつゴール節の答を見つけるために, OR 分岐で全数探索をすることが必要である。Prolog-ELF では一般にこの探索は BEST(1) モードで実行されるが, 述語 MAXCL, DMAXCL の使用によってもこの最大の真理値をもつ答の探索の実行を指定することができ

```
>-load(demo ).
>+very(*cl)-value(*cl,*x)-sqr(*x,*y)-usevalue(*y).
success
>-old(60).
0.8500:-old(60).
>-very(-old(60).).
0.7225:-very(-old(60).).
```

図3 述語 VALUE の使用例

(>はシステムのプロンプト記号) 図2の定義が仮定されている。

Fig. 3 An example of the usage of VALUE. (> is a system prompt.) The definition of Fig. 2 is assumed.

る。この述語は次の理由によりサブゴール・ノードでも使用される。たとえば, もし

0.9: +R(*x)-P(*x)-Q(*x).

0.9: +P(a).

0.7: +P(b).

0.6: +Q(a).

0.8: +Q(b).

が定義されていたとすると, 次のようになる。

$T(R(a)) = \min(0.9, \min(0.9, 0.6)) = 0.6$

$T(R(b)) = \min(0.9, \min(0.7, 0.8)) = 0.7$

これは, もし最大の真理値を有する $R(*x)$ の具体値 (この場合は $R(b)$) を答とすると, この具体値は必ずしもサブゴール・ノード $P(*x)$ が最大真理値となる場合ではない。多くの応用では, サブゴールのリテラルも最大の真理値になるようなユニフィケーションに固定したいことがあり, これは述語 MAXCL, DMAXCL の使用によって指定できる。

たとえば上記の定義に次の節を加えると,

0.9: S(*x)-maxcl(P(*x).)-Q(*x).

システムは次のように応答する。

>-maxcl(S(*x).).

0.6: maxcl(S(a).).

MAXCL は非決定的な述語であり, DMAXCL は決定的な述語である。すなわち, 同じ最大の真理値をもつ答が2個以上得られたときに, MAXCL の場合はバックトラック時に先に使われたのと別の答が使用されるのに対し, DMAXCL の場合はバックトラック時に失敗 (fail) となる。

(7) NOT:

Prolog-ELF でも述語 NOT を使用でき, 通常のスレッシュールド設定時には引数として与えられた節に対する答が一つでも存在するときには失敗 (fail) し, 答が存在しないときに成功することは通常の Prolog と同じである。ただし, もしスレッシュールド値が 0.5 以下に設定されていると (例外的な場合として設定できる), NOT の実行中にユニフィケーションの結果が残ることがある。たとえばスレッシュールドの設定値が 0.2 とされており, $P(*x)$ に対する上限真理値 (この場合は下限真理値でなく上限をとる) 付き答が 0.7: $P(a)$ である場合, not ($P(*x)$) は下限真理値 $0.3 (= 1 - 0.7)$ で成功することになり, *x は a にユニファイされることになる。しかし, この機能は現バージョンではインプリメントされていない。

```

; 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,30000000)-/.
+rich2(*yen)-gt(*yen,15000000)-/
  -minus(*yen,1.5e7,*x)-rdiv(*x,1e8,*y)
  -times(3,*y,*z)-plus(0.5,*z,*w)
  -usevalue(*w).

; Income ;
+income(Yamada,20000000).
+income(Sato,25000000).
+income(Nomura,8000000).
+income(Kobayashi,40000000).

; President ;
+president(Suzuki).
+president(Kobayashi).

```

(a) データベース
database

```

Prolog-ELF version 2.1a      6/12/84
written by N. Kanai, Ishizuka lab.

type help for "-help.", exit for "-exit.".
>-load(house_).
>-have_a_second_house(*who).
1.0000:-have_a_second_house(Kato).
>-best(5).
1.0000:-best(5).
>-have_a_second_house(*who).
1.0000:-have_a_second_house(Kato).
0.9000:-have_a_second_house(Kobayashi).
0.8000:-have_a_second_house(Sato).
0.7000:-have_a_second_house(Suzuki).
0.6500:-have_a_second_house(Yamada).

```

(b) 実行
execution

図4 論理データベースへの応用

Fig. 4 An application to a logical database.

4. ファジィ論理データベースへの応用例

Prolog-ELF の効用を示すために、直接的な応用例として、ファジィルールとファジィ集合概念を含む論理データベースを図4に示している。ここでは別荘(a-second-house)をもつであろう人が、事実とヒューリスティックなルールから成る知識(データ)ベースから検索される様子が示されている。

筆者らにより Prolog-ELF を用いてファジィ・プログラム・システムを含む幾つかの応用プログラムが作成されている。

5. むすび

不確実性やファジィ性を含む知識型システム用の基礎言語として、ファジィ論理と他の有用な機能を組み入れて作成した Prolog-ELF について記した。

Prolog-ELF では BEST(n) モード、あるいは MAXCL, DMAXCL を実行すると、OR 分岐で全数探索が必要なことから、速度が遅くなるという問題がある。この問題の現状での解決法は、必要以外の場所でこれらのモード、述語を使用しない、またはスレッシュホールド値を適当に高くすることである。今後のバージョンでは探索効率向上の機構を組み入れることを考えているが、究極的には逐次形の探索制御でなく、第5世代コンピュータで計画されているような並列プロセッサによる並列探索によって解決されることになろう。Prolog-ELF はメタ述語の導入によるファジィ述語論理環境における知識管理機構の組み入れも予定し

ている。

謝辞 Prolog-ELF の基本部分は Prolog/KR の作成者である中島秀之氏(電総研)のプログラムを参考にしてインプリメントしたことを記し、謝意を表します。本研究は文部省特定研究「多元知識情報の知的処理と統合化に関する研究」の援助によるものである。

参考文献

- 1) 石塚 満: 不確かな知識の取り扱い, 計測と制御, Vol. 22, No. 9, pp. 774-779 (1983).
- 2) Shortliffe, E. H.: *Computer-Based Medical Consultation: Mycin*, American Elsevier, New York (1976).
- 3) Duda, R. O., Hart, P. and Nilson, N. J.: *Subjective Bayesian Methods for Rule-Based Inference Systems*, National Computer Conf. (1976).
- 4) Weiss, S. M., Kulikowski, C. A. et al.: *A Model-Based Method for Computer-Aided Medical Decision-Making*, *Artif. Intell.*, Vol. 11, pp. 145-172 (1978).
- 5) 石塚 満: 建築物被害査定のエキスパート・システム, 情報処理学会論文誌, Vol. 24, No. 3, pp. 357-363 (1983).
- 6) Ishizuka, M.: *Inference Methods Based on Extended Dempster & Shafer's Theory for Problems with Uncertainty/Fuzziness*, *New Generation Computing*, Vol. 1, No. 2, pp. 159-168 (1983).
- 7) Shapiro, E. Y.: *Logic Programs with Uncertainties: A Tool for Implementing Rule-Based Systems*, Int. Joint Conf. on Artif. Intell. (1983).

- 8) Dubois, D. and Prade, H.: *Fuzzy Set and Systems: Theory and Applications*, Academic Press, New York (1980).
- 9) 石塚 満: Dempster & Shafer の確率理論, 電子通信学会誌, Vol. 66, No. 9, pp. 900-903 (1983).
- 10) 金井直樹, 石塚 満: ファジィ論理を組み込んだ Prolog-ELF, 情報処理学会知識工学と人工知能研資, No. 34-4 (1984).
- 11) Zadeh, L. A.: PRUF—A Meaning Representation Language for Natural Language, *Int. J. Man-Mach. Stud.*, Vol. 10, pp. 395-460 (1978).
- 12) Umamo, M., Mizumoto, M. and Tanaka, K.: FSTDS System: A Fuzzy-Set Manipulation System, *Inf. Sci.*, Vol. 14, pp. 115-159 (1978).
- 13) Zadeh, L. A.: Fuzzy Logic and Approximate Reasoning, *Synthese*, Vol. 30, pp. 407-428 (1978).
- 14) Lee, R. C. T.: Fuzzy Logic and the Resolution Principle, *J. ACM*, Vol. 19, No. 1, pp. 109-119 (1972).
- 15) 向殿政男, 増沢秀穂: Fuzzy 論理における導出形の性質について, 電子通信学会論文誌, Vol. J 66-D, No. 7, pp. 796-803 (1983).
- 16) Whalen, T. and Schott, B.: Issues in Fuzzy Production Systems, *Int. J. Man-Mach. Stud.*, Vol. 19, pp. 57-71 (1983).

(昭和59年12月20日受付)

(昭和60年12月19日採録)



金井 直樹 (正会員)

昭和35年生。昭和58年東京大学工学部電子工学科卒業。昭和60年同大学院修士課程修了。同年日本アイ・ビー・エム(株)入社。現在、同社サイエンス・インスティテュートに勤務。知識工学, 特にプランニング, エキスパート・システムなどの研究に従事。



石塚 満 (正会員)

昭和23年生。昭和46年東京大学工学部電子工学科卒業。昭和51年同大学院博士課程修了。工学博士。同年 NTT 横須賀研究所勤務。昭和53年より東京大学生産技術研究所助教授。昭和55年より1年半の間米国 Purdue 大学客員準教授。知識工学, 画像理解の研究を行っている。電子通信学会, 画像電子学会, IEEE, AAI の会員。