あいまい概念を扱う知識表現における 目標指向型推論[†]

吉 江 修*・藤 村 茂**・宮 地 裕 樹*・秋 月 影 雄*

On Top-Down Reasoning Using Knowledge Representation Dealing with Ambiguous Concepts

Osamu YOSHIE*, Shigeru FUJIMURA**, Hiroki MIYACHI* and Kageo AKIZUKI*

Ambiguous concepts play an important role in the reasoning that appears in our mental activity. Many expert systems have been developed as models of such kind of activities recently, but most of them cannot deal with ambiguous expression directly. This leads to difficulty of the knowledgebase construction and the input of ambiguous facts.

So we propose the knowledge representation dealing with ambiguous concepts and top-down reasoning method using it. We're able to treat ambiguous knowledge directly and get a satisfactory conclusion from the fewest facts by this proposed method.

The efficiency of this method is demonstrated via an example.

Key Words: expert system, frame, fuzzy reasoning

1. ま え が き

われわれが行っている知的活動にあらわれる推論作業において、あいまいな概念から得られる情報は大きく、重要な役割を果たしている。特にプラントの故障診断、医学における患者の診断などにおいては、感覚的なあいまいな概念や、取得したデータから対応づけられるあいまいな概念というものは重要で、それらによって推論がなされ異常原因を導き出している。

今日までにこのような知的活動のモデルとしてのエ

キスパートシステムが多方面にわたって開発されているが^{1),2)}、事実をその推論に用いられる知識にあてはめる際、ただ文字処理的なパターンマッチングを行い、あいまい性の表現はその事実の確信の得られる程度として用いられているものがほとんどで、推論に用いられる知識に直接あいまい性の表現が与えられているものは少ない。そのために、専門家による知識ベースの作成、利用者によるあいまい事実の入力は直接的な表現がとれず、あいまい性の表現が付加的なものになってしまい、推論効率の向上、すなわち探索事実の減少を図ることはできない。

エキスパートシステムは、知識ベースの断片的な知識を推論機構によって組合せ、見かけ上大きな知識として多様な問題に対処しようというもので、あいまい性を伴う問題を扱う場合にも、知識が正しければ正しい回答が得られるものでなければならない。一般に、あいまい性を伴う問題を部分問題に分割し階層的に記述すると、その関係は AND/OR/COMB グラフで表わされる³⁾。推論によって十分合理性のある回答を得るためには、このグラフ上で、先行する推論あるいは観察された証拠としてある部分問題の確実性測度がすでに求められているときに、より上位の問題の確実性測度を適切な方法で推論し決定することが大切である。このうち AND, OR 関係に関しては常識に従って容易に決定でき、それぞれある確実性測度上で min, max 演算を採用できる。したがってこれらの関係とともに COMB 関係への対処の仕方も決定されれば、確実性測度は階層的推論ネットワーク中を伝搬することができる。そして最終的に目標ゴールにいたる確実性測度が求められ、これによって合理的な答を決定できる。

[†] 昭和 60 年電気学会全国大会で発表 (昭 60・4)

* 早稲田大学理工学部 東京都新宿区大久保 3-4-1

** 横河北辰電機(株) 武蔵野市中町 2-9-32

* Faculty of Science and Engineering, Waseda University, Shinjuku-ku, Tokyo

** Yokogawa Hokushin Electric Corporation, Musashino

(Received January 17, 1986)

(Revised May 8, 1986)

以上の点を踏まえ、本論文ではあいまい概念を表現するために、Fuzzy 集合を用いたパターンマッチングを行う知識表現形式による、目標指向型推論機構を提案する。そこでは Fuzzy 集合を用いたために結合関係を AND/OR に縮退することができ、推論効率の向上を行うことができる。これは COMB 関係における部分問題の独立性の仮定が通常は満たされず、Fuzzy 推論を用いた場合、その確実性測度はそれぞれの部分問題の測度の min 値と max 値の間の値となり⁴⁾、COMB 関係を OR 関係に縮退するのが好ましいからである。また知識ベース上の規則に対して、あいまい概念が直接蓄積できる形となる。

以下、2 章ではあいまい概念をとり扱うための知識表現形式を、3 章ではこれを基に必要最小限の事実から効率良く合理的な結果を導き出す目標指向型推論方法について述べる。またゲームの木の探索法として有名な α - β 法⁵⁾との推論効率の比較を行う。さらに 4 章において実行例として冷凍機の故障診断をとりあげ、実際にどのように推論が行われるかを示す。

2. あいまい概念をとり扱うための知識表現形式

いま、われわれが知識としてもつルールとそのルールの項との関係が、Fig. 1 に示すような AND/OR 木で表わせるものとする。図中の二重円はルールを表わし、その中に書かれているものはそのルールの番号である。その他の円はルール中に現れる項であり、その中に書かれているものは項の名前を表わし、項と事実との適合度がその横に書かれている。トップレベルのノードは、目標とするノードのどれかが存在していることを表わし、またその下のノード 10, 20 は、目標とするノード $t1$, $t2$ が存在すればトップレベルのノード $t0$ が成立することを表わすルールであり、どちらも仮想的なノードである。そして $t1$, $t2$ が目標となるノードで、この例では 2 個であるが、実際にはも

っと多くのノードが存在する。

このような知識を単にプロダクションルールの形式で表現したのでは、推論手順において事実の探索順序に関連性がうすくなり、またある事実を知ったときにその結果を推論手順の決定に反映させることは困難となる。そこでこれらの問題点をさけるため、つぎのような知識表現形式を提案する。すなわち、ルールに現れる項どうしを有機的に結びつけるために、ルールの項は単に AND/OR 木のノードとして表現しておき、そのノードにフレーム表現を採用することによりノード間での情報の受け渡しができるようにし、推論手順を制御することにする。このようにルールの項どうしの関係をルールの外に表現することにより、人間の知的活動の振舞いに近い推論が可能となる。以下では、その知識表現の詳細について述べる。

(1) AND/OR 木のノード関係

AND/OR 木の状態は、つぎのような形で記憶しておく。

```
data (node_name, [c_node 1, c_node 2, ...,
c_node m, g]).
```

第 1 要素はノードの名前を表わし、第 2 要素はその子ノード、すなわちルールの前提部を表わす。リストの内容はその要素の AND を意味しており、たとえば Fig. 1 のノード $t1$ については

```
data (t1, [t3, t4, t5, g]). data ([t1, t6, t7, g]).
```

と記述される。リストの最後に g のあるものは、その第 1 要素が目標ノードであることを示している。

(2) ノードフレーム

ノードフレームは、つぎのような形式をとるものとする。

```
node (node_name, default_value,
n_v(n_term_name, default_define, n_msv),
f_v(f_term_name, f_msv)).
```

第 1 要素はノードの名前で、AND/OR 木におけるノードとの対応をとるためのものである。第 2 要素はそのノードのデフォルトの適合度で、推論手順を決定するために用いられる。第 3 要素はそのノードの適合度を得るときに参照され、項の指定および適合度が得られたときの他のノードの default-value の変更を行うためのものである。第 4 要素は簡易的な推論のために用いられるもので、たとえば感覚的なあいまい概念によってノードの適合度を変えて推論手順の変更を行い、計算データの取得のような手間のかかるパラメータ値の探索は、より必要のある場合にのみ行わせることができる。この要素は必要に応じて用いるものとする。

Fig. 1 An example of AND/OR tree

(3) ノードの適合度の決定

ノードの適合度は、直接ノードによって決定せず、ノードフレームを介してつぎの項

term(term_name, parameter, term_value).

と事実との適合度を求めることにより得るものとする。第2要素のパラメータとしては、数値型・多値近似型・Yes-No型の3種類を考え、おのおののFuzzy集合的なパターンマッチングの方法を以下に示す。

(3.1) 数値型

パラメータが数値型の場合には、term_valueとして数値あるいはFuzzy集合のメンバシップ関数を扱うことができる。いま、あるノードに注目しその適合度を得たいときに、そのノードフレームがつぎの項を指定しているとする。

term(term_name 1, x, gt 20).

gt 20 (greater than 20) に関してはあらかじめメンバシップ関数を **Fig. 2** のように与えておくことにする。パラメータ x の値として、ある事実からたとえば18が得られると、それに応じて適合度0.9が得られる。この例ではルールの項と事実とのパターンマッチングのうえで境界値 $x=20$ があいまいで、その周辺のメンバシップ関数の振舞いはわれわれの感覚に矛盾しないように決定すべきものである。

(3.2) 多値近似型

多値近似型のパラメータを導入することにより、たとえば

very short, short, normal, long, very long

のような感覚的表現を扱うことができる。

term(term_name 2, y, flong).

パラメータ y が多値近似型で、term-value flong に対して **Table 1** のようなメンバシップ値が定義されているとする。このときパラメータ y の入力促進に

Fig. 2 Membership function of gt 20

Table 1 Membership value of flong

y	
verylong	1.0
long	1.0
normal	0.8
short	0.2
veryshort	0.0

応じて normal が入力されたとすると適合度0.8が得られる。

(3.3) Yes_No 型

term_value として yes あるいは no をとり、入力されたパラメータ値と一致したときに適合度1、不一致の場合適合度0とする。

(4) パラメータ宣言

ルールの AND/OR 木において、終端ノードに対応する項にあらわれるパラメータと、推論の途中であらわれるパラメータの2種類を考える。前者はさらに、推論中に必要に応じて質問をして得る ask_first 型と、推論を行う前にあらかじめ利用者に質問しておく initial_data 型に分けられる。また後者は infer 型と呼ぶことにする。パラメータ型の宣言はつぎのように行う。

param(y, a, apx, [very_short, short, normal, long, very_long]).

param(z, i, y_n, _). param(w, n, flt, _).

第1要素はパラメータ名であり、第2要素については a が ask_first 型、 n が infer 型、 i が initial_data 型を示す。また第3要素はそのパラメータが apx のとき多値近似型、 y_n のとき Yes_No 型、flt のとき数値型であることをそれぞれ表わしている。第4要素は多値近似型の場合のみ必要で、そのとるべき感覚名が与えられている。

initial_data 型のパラメータは推論を始める前に入力されるべきもので、この入力によりノードフレームの default_value が書きかえられ、そのパラメータの入力の違いによって推論手順が異なったものとなる。

パラメータフレームはつぎのように表わされ、おのおののパラメータについてその値を得る方法が記されている。

frame(w, [value, if_needed(u/2.0)]).

frame(u, [value, if_needed(ask_value)]).

u のパラメータフレームは質問して値を得ることを示しており、if_needed(ask_value) によってそのフレーム名の質問文を他のファイルから呼びこみ、CRT 上で入力促進することを表わしている。 w のパラメータフレームには、他のパラメータフレーム u の値からの計算によって求められることが if_needed(演算式) によって記されており、この部分はネスティング可能であるように定義されている。

(5) メンバシップ関数の定義

数値型・多値近似型パラメータのFuzzy集合的なパターンマッチングに必要なメンバシップ関数の定義はつぎのように行う。

```

fdata(y, flong, [r(very_short, 0.0), r(short, 0.2),
r(normal, 0.8), r(long, 1.0),
r(very_long, 1.0)]).
fdata(u, abig, [r(0.0, 0.8, zero), r(0.8, 1.3, fl),
r(1.3, 100.0, one)]).
func(zero, -, 0.0): -!. func(one, : _1.0): -!.
func(f1, I, O): -O is (I-0.8)/0.5.

```

第1要素にはパラメータ名, 第2要素には term-value, すなわちメンバシップ関数名が書かれている。第3要素は, パラメータが多値近似型の場合, そのとりうる感覚名に対するメンバシップ値のリストになっている。また数値型の場合には, そのとりうる範囲において区分分けを行い, おおのこの区分とともにそのときのメンバシップ関数を与える関数名のリストになっている。その関数については func (関数名, 数値, メンバシップ値) として PROLOG による記述がなされている。

3. 推論機構

前章では, 知識ベース上のルールに対してあいまい概念を直接蓄積できるような知識表現形式について述べた。ここではまず Fig. 3 に示すような AND/OR 木のノード関係を記憶しておき, そしてそれぞれのノードにフレーム表現を用いてノード間での情報の受け渡しを可能にし, default-value により推論手順を制御するものであった。本章ではこの推論手順の制御の詳細について述べ, 本推論法がゲームの木の探索法として有名な α - β 法と比較してより効率が高いことを示す。簡単のために Fig. 3 における終端ノードの適合度は図のように与えられるものとする。実際にはこれらの適合度は, 一番高い適合度をもつ目標ノードを探索する過程で, 必要に応じて求められるものである。したがってエキスパートシステム構築の立場としては, 必要最小限の項と事実のパターンマッチングに

Fig. 3 AND/OR tree and reasoning result.

The number with underline represents the initial data of node frame default value, and the number represents membership value. $\times$ and $\circ$ mean that the node is traversed by α - β method and proposed method respectively

よって目標の項が得られるように, 有望な部分, すなわち適合度の高い部分から推論を行っていくことが望ましい。

3.1 推論方法

推論方式は目標指向型とし, 常に適合度が最も高いノードについて展開していく方法をとる。Fig. 3 の例について述べると, まずすべての目標ノードに注目する。t1, t2 の子ノードは

t1: [t3, t4, t5]... (ア) t1: [t6, t7]... (イ)

t2: [t8, t9, t10]... (ウ)

であり, t1, t2 の default-value はともに 1.0 で等しいので (ア)~(ウ) のどれに注目してもよいが, ここでは AND/OR 木を記憶している data 文の順番によって, まず (ア) に注目するものとする。すると t3 の適合度 0.8 が得られ, この場合のノード t1 の適合度はたかだか 0.8 となってしまう。そこでより適合度が高い可能性のある (イ) (ウ) のうち (イ) に注目する。このように推論を続けていくが, その手順をまとめると Fig. 4 に示すフローのようになる。ただし data1 は現在注目しているノードを, また data2 は展開したノードを記憶するためのもので,

```

data1 (node_name, [p_node1, ..., p_node m],
[c_node1, ..., c_node n],
[dv1, ..., dvn], dv, min).

```

data1 の第1要素は注目ノードの名前を, 第2要素はその親ノードを示す。これまでは AND/OR 木についてのみ述べてきたが, 終端ノードのあるグラフであれば, 第2要素の内容が複数になるだけで以下の議論には差しつかえない。第3要素は子ノードの名前のリストとなっていて, 注目しているノードが目標ノードの

```

select <
| select data1 that has the largest 6th element;
| if the 3rd element is [o] ... goal;
| apply_data
| if the 3rd element = [ ] ... make new data1 from its parent node
data2 in the following manner: remove the heads of the 3rd and
4th element of data2, the 5th element := the 5th element of old
data1, the 6th element := minimum of the values of the 4th and 5th
element;
remove all descendants of the node corresponding to new data1;
else ... pick up the head node of the 3rd element;
if we can apply 'apply_node' and the return value < 0 ...
if we can apply 'apply_node' and the return value > 0 ...
change data1 in the following manner: remove the heads of
the 3rd and 4th element, the 5th element := minimum of the
value of the 5th element and the return value, the 6th element
:= minimum of the values of new 5th element and new 4th element;
else ... change data1 to data2 by removing its 6th element and
add data1 that has the head of the 3rd element of new data2
as its parent, where the 5th element := the 5th element of
old data1, the 6th element := minimum of the values of new
5th element and 4th element;
| apply_node
| if it is a terminal node ...
| if f_v value is not defined yet ... define f_v value and make it
new default_value, rewrite data1 and data2 corresponding to
the node, return -1*(f_v value);
| if only f_v value is defined ... define n_v value and change
default_value, do default, define(dd), change data1 and data2
related to those nodes, return n_v value;
| else ... return default_value;
| else ... fail;

```

Fig. 4 Flow of reasoning process

場合はその最後に g を加えておく。第4要素はその子ノードに対応して、その時点までに得られているノードフレームの `default_value` のリストとなっている。そして第5要素はその時点までに得られているそのノードの適合度、すなわちそのノードが属している目標ノードのとりうる適合度の上限値を示している。第6要素はその値と第4要素のリスト中の値との `min` の値であり、ノードの展開はこの値の大きいものを優先的に行う。そして展開されたノードは、`data 1` の第6要素をとり除いた形をもつ `data 2` として記憶される。

3.2 α - β 法との推論効率の比較

前節で述べた推論法と α - β 法によって探索されたノードの比較を Fig. 3 に示す。提案した推論法が効率の点で優れていることがわかるが、このことは木の構造に依存しない。すなわち α - β 法が原則的に縦型探索であるのに対し、本探索法は常に木全体の中から適合度が最も高い可能性のあるノードを優先的に展開し、探索の打ち切りに関しては α カット・ β カットの両方を行っているからである。また α - β 法では、探索を行った後つぎに高い適合度をもつ目標ノードを得ようとする、 α カットにより探索途中であるノードに関しては情報が残っていないため、探索を繰返さなければならない。しかし本探索法では、適合度の最大値が求まったときに、他の目標ノードに関してその時点までの推論で得られている適合度がわかり、そのノードの適合度の上限値を示すことができる。そのうえ、適合度の最大値が得られた後も注目しているノードを記憶しているので、その得られたノードの情報をとり除くとそのまま探索は続けられ、順次大きい順に適合度を求めることができる。そのために α - β 法では α と β の値だけを記憶しておけばよいのに対して、本探索法では注目ノード、展開ノードを記憶しておく領域が必要であるという欠点はあるが、上記のようにマン・マシン系として優れている面があるうえ、推論効率の点で勝っている、注目ノード、展開ノードの記憶領域を確保できる状況下では、本手法が有効であるといえる。

4. 実行例

本章では今までに述べた知識表現形式、推論方式を用いて、実際に冷凍機の故障診断を行った例を示す。診断に用いた知識は Fig. 5 に示すような AND/OR 木で与えられるものとし、故障原因としては「冷却水量が少ない」(ノード1)、「冷凍装置に空気が混入している」(ノード2)の二つを考える。以下に推論がどのように行われるかについて詳しく述べる。

まずすべての目標ノードに注目し `data 1` とする。

```
data 1 (1, 0, [9, 10, 0], [1.0, 1.0], 1.0, 1.0).
```

```
data 1 (2, 0, [9, 17, 0], [1.0, 1.0], 1.0, 1.0).
```

そしてノード1に注目し、子ノードの先頭の要素ノード9についてノードフレームを参照する。その結果非終端ノードであることがわかり、ノード1について展開して `data 2` とし、ノード9を `data 1` として加える。

```
data 1 (9, 1, [13], [1.0], 1.0, 1.0).
```

```
data 1 (9, 1, [11, 12], [1.0, 1.0], 1.0, 1.0).
```

```
data 1 (2, 0, [9, 17, 0], [1.0, 1.0], 1.0, 1.0).
```

```
data 2 (1, 0, [9, 10, 0], [1.0, 1.0], 1.0).
```

最上位の `data 1` に注目し、同様にノード9について展開する。`data 1` の挿入位置は第6要素の比較によって決めるが、同じ場合はそれらの一番上に挿入することにより縦型探索に近づけている。

```
data 1 (13, 9, [14, 16], [1.0, 1.0], 1.0, 1.0).
```

```
data 1 (9, 1, [11, 12], [1.0, 1.0], 1.0, 1.0).
```

```
data 1 (2, 0, [9, 17, 0], [1.0, 1.0], 1.0, 1.0).
```

```
data 2 (1, 0, [9, 10, 0], [1.0, 1.0], 1.0).
```

```
data 2 (9, 1, [13], [1.0], 1.0).
```

ここでノード14のノードフレームを参照すると終端ノードを表していて、`n_v` に関して項 `n 14` とのパターンマッチングを促進している。

```
node (14, 1.0, n_v (n 14, [ ], nd)).
```

```
term (n 14, stop, yes).
```

項 `n 14` は「起動させてもすぐに停止してしまう」かどうか、その適合度を得るためのもので、`parameter stop, term-value yes` であるので、パラメータフレーム `stop` を参照する。

```
frame (stop, [(value, if_needed(ask_value))]).
```

すると `ask_value` なので、そのパラメータの値について質問し値を得る。そしてパラメータの型の宣言

```
param (stop, a, y_n, _).
```

を参照し Fuzzy 集合的なパターンマッチングを行う。

```
stop soon ? *** yes. <<<apnode 14 1.0>>>
```

こうして適合度 1.0 が得られ、`data 1` およびノード14のノードフレームはつぎのように更新される。

```
data 1 (13, 9, [16], [1.0], 1.0, 1.0).
```

```
data 1 (9, 1, [11, 12], [1.0, 1.0], 1.0, 1.0).
```

```
data 1 (2, 0, [9, 17, 0], [1.0, 1.0], 1.0, 1.0).
```

```
data 2 (1, 0, [9, 10, 0], [1.0, 1.0], 1.0).
```

```
data 2 (9, 1, [13], [1.0], 1.0).
```

```
node (14, 1.0, n_v (n 14, [ ], 1.0)).
```

同様にしてノード16に関して「冷凍装置の凝縮圧力 h_p 高圧圧力開閉器の設定値を上回っている」かどうか

Fig. 5 An example of the execution of proposed method

か、 $hp=10.9$ の場合について適合度 0.95 が得られ、
 node(16, 1.0, n_v(n 16, [dd(11, 1.0-self)], nd)).
 term(n 16, hp, over).
 frame(hp, if_needed(ask_value)).
 param(hp, a, flt, _).
 Input hp ! ***10.9.
 <<<apnode 16 0.949999>>>
 ノード 16 のノードフレームはつぎのように更新される。またその際 default_define によってノード 11 のノードフレームの default_value は 0.05 に書きかえられる。

```

node(16, 0.949999, n_v(n 16, [dd(11, 1.0-self)],
0.949999)).
node(11, 0.05, n_v([ ], nd)).
  
```

このとき注目・展開ノードはつぎのようになるが、ノード 2 の子ノード 9 についてはすでに展開されている。

```

data 1 (2, 0, [9, 17, g], [1.0, 1.0], 1.0, 1.0).
data 1 (13, 9, [ ], [ ], 0.949999, 0.949999).
data 1 (9, 1, [11, 12], [0.05, 1.0], 1.0, 0.05).
data 2 (1, 0, [9, 10, g], [1.0, 1.0], 1.0).
data 2 (9, 1, [13], [1.0], 1.0).
  
```

て、同じ展開を行う必要はない。またその展開は親ノード 1 についてなされていて、そのときの適合度は現在の適合度より勝っている。そこで今回の展開は親の情報さえ残しておけばよいのでつぎのように展開しておく。

```

data 1 (13, 9, [ ], [ ], 0.949999, 0.949999).
data 1 (9, [2, 1], [11, 12], [0.05, 1.0], 1.0, 0.05).
data 2 (1, 0, [9, 10, g], [1.0, 1.0], 1.0).
data 2 (2, 0, [9, 17, g], [1.0, 1.0], 1.0).
data 2 (9, [2, 1], [13], [1.0], 1.0).
  
```

ノード 13 に関しては子ノードをすべて展開しおわ

ているので、data 2 のうち親ノード 9 のノードで、第 3 要素の先頭がノード 13 のものを data 1 として挿入する。

```
data 1 (9, [2, 1], [ ], [ ], 0.949999, 0.949999).
data 1 (9, [2, 1], [11, 12], [0.05, 1.0], 1.0, 0.05).
data 2 (1, 0, [9, 10, 9], [1.0, 1.0], 1.0).
data 2 (2, 0, [9, 17, 9], [1.0, 1.0], 1.0).
```

同様にノード 1, 2 について data 2 から data 1 とする。ただし子ノード 11, 12 をもつ data 1 については、ノード 16 の探索の際に default_define によりノード 11 の default_value が下げられ推論手順が落とされているため、現在の最高の適合度をもつノードに対し木を逆のぼるときに情報を保持しておく必要がないので取り除く。

```
data 1 (1, 0, [10, 9], [1.0], 0.949999, 0.949999).
data 1 (2, 0, [17, 9], [1.0], 0.949999, 0.949999).
```

つぎにノード 10 については f_{lv} が付いているので、その適合度を得る。項 f₁₀ は「冷却水の入口温度と出口温度の差 ciotf が大きい」というルールの前提部と事実とがどの程度適合するかを求めるためのもので、「大きい」という感覚的であいまいな概念を用いてルールが直接記述可能となっている。これに対しユーザ側にも「温度差はなるべく同じくらい」という、あいまいな表現による入力に許されている。

```
node (10, 1.0, n_v (n 10, [ ], nd), f_lv (f 10, nd)).
term (f 10, ciotf, abig).
frame (ciotf, if_needed (ask_value)).
param (ciotf, a, apx, [same, big, vbig]).
```

How is coolant temp difference between inlet and outlet ?

```
*** same. <<<apnode 10-0.099999>>>
```

こうして適合度 0.1 が得られ

```
node (10, 0.099999, n_v (n 10, [ ], nd), f_lv (f 10,
0.099999)).
data 1 (2, 0, [17, 9], [1.0], 0.949999, 0.949999).
data 1 (1, 0, [10, 9], [0.099999], 0.949999,
0.099999).
```

のように更新される。まったく同様にしてノード 17 について適合度 0.83 が得られ、この値とノード 2 の data 1 の第 5 要素との min をとり、第 6 要素は 0.83 となる。この値は他の data 1 の第 6 要素に比べ最

も大きいので、ノード 2 が現在までの推論で最も適合度が高く、故障原因としてまず「冷凍装置に空気が混入している」ことが考えられると推論された。

Input difference between temp of coolant at outlet and condensation !

```
***7.5. <<<apnode 17 0.833333>>>
```

```
*****
```

Membership value of 2 is 0.833333.

```
*****
```

Do you want to continue this reasoning ?

```
***yes.
```

さらに今得られたノード 2 の data 1 を取り除くことにより、Fig. 5 に示すように推論を継続することができる。このようにつぎに適合度の高い目標ノードの探索を即座に行えることは本探索法の一つの長所であるが、反面前述したように data 1, data 2 として注目・展開ノードの記憶領域を確保することが必要である。

5. あとがき

知識ベース上のルールに対してあいまい概念を直接蓄積できるような知識表現形式と、そのうえでの効率の良い目標指向型推論について述べた。本論文における手法は、必要最小限の質問によりルールの項と事実との間に Fuzzy 集合的なパターンマッチングを行って適合度を得、推論の結果適合度の大きい結論を導いていくという点に特徴がある。

現在実際の推論においては、ノードフレームの default_value はすべて 1.0 として推論を開始しているが、この初期値の設定は推論効率を左右する問題であるので、今後の課題である。

参 考 文 献

- 1) E. H. Shortliffe: Computer-Based Medical Consultations: MYCIN, American Elsevier (1976)
- 2) R. O. Duda, P. Hart and N. J. Nilsson: Subjective Bayesian Methods for Rule-Based Inference Systems, Nat. Computer Conf. (1976)
- 3) M. Ishizuka, K. S. Fu and J. T. P. Yao: Inference Procedure with Uncertainty for Problem Reduction Method, Information Sciences, **28**, 179/206 (1982)
- 4) M. Ishizuka, K. S. Fu and J. T. P. Yao: A Rule-Inference Method for Damage Assessment, ASCE Conv., St. Louis, 81/502 (1981)
- 5) Avron Barr, et al., 田中, 淵監訳: 人工知能ハンドブック I, 103/106, 共立出版 (1983)