認知的属性とファジィ理論を用いた意味フレーム決定システム

Semantic Frame Determination System

Using Cognitive Attribute and Fuzzy Logic

馬野 元秀 [1] ○後藤 聡 [1] 船山 仲他 [2] 宇野 裕之 [1]

Motohide UMANO　Satoshi GOTO　Chuta FUNAYAMA　Yushi UNO

野田 尚史 [2] 佐藤 浩 [1] 中村 直子 [2]

Hisashi NODA　Hiroshi SATOH　Naoko NAKAMURA

1: 大阪府立大学 総合科学部 数理・情報科学科
Department of Mathematics and Information Sciences
College of Integrated Arts and Sciences, Osaka Prefecture University

2: 大阪府立大学 総合科学部 総合言語文化学科
Department of Language and Culture
College of Integrated Arts and Sciences, Osaka Prefecture University

1. はじめに

ファジィ集合論は、もともと自然言語の意味を表現するために提案された。しかし、ほとんどの研究ではファジィルールを用いて数値の計算を行なっているだけである。

L.A. Zadeh は、ファジィ集合、ファジィ修飾語、ファジィ接続語、ファジィ限定子などを用いて自然言語の意味を表現する方法として、PRUF やテスト・スコア意味論を提案した [1],[2],[3]。

また、我々は以前に簡単な日本語による質問応答システムを試作した [4]。このシステムでは、箱や花や机に関する重さや大きさや長さをファジィ集合を用いて述べた後に、ファジィ集合を用いた質問を行なうと、一致度を調べて、「真である」、「たぶん真である」、「偽である」、「たぶん偽である」、「どちらともいえない」に分類し、それぞれに応じた答を返すことができた。また、最近、藤本・菅野らは機能言語理論とファジィ理論の結合について提案している [5]。

人間が自然言語の表現を実際に聞いたり見たりして、その意味を理解する過程では、さまざまなファジィ的な知識処理を行なっていると考えられる。本論文では、自然言語理解の過程でファジィ集合演算を用いることを試みる。具体的には、認知的属性という考え方を用い、ファジィ演算により意味フレーム (主語や目的語などがどのような意味的役割を果たしているかを表わす枠組み) を決定する新たな方法を提案し、試作したシステムの実行例を示す。このシステムはファジィ・フレーム・システム [6] を知識ベースとして用いている。これにより階層的な辞書の記述が可能である。さらに、このシステムでは論文 [7] で提案されている、節と節を越えた述語—項関係を成り立たせる方法を用いることにより、前の文に含まれている単語を、意味フレーム決定に加える機能を備えている。

2. 考察の対象とする述語

文中の名詞句と述語の意味的関係は基本的には統語構造によって規定される。日本語の場合には、格助詞によって主語や目的語が識別され、述語の関係が表わされ、さらに、それに応じて述語の意味フレームのどの主題役割 (thematic role) にどの名詞句が対応するかが決まる。しかし、統語情報だけでは決まらない場合もある。

ここで取り上げる述語は、「飾る」、「塗る」、「埋める」などの、いわゆる場所格交替 (locative alternation) を可能とする動詞である。このような動詞は、「テーブルを花で飾る」、「テーブルに花を飾る」というような 2 種類の構文をとれる。前者では「を」が付いた名詞句は “飾る場所” を、「で」

が付いた名詞句は “飾る手段となる移動物” を表わしている。後者では、「に」が “飾る場所” を、「を」が “飾る手段となる移動物” を表わす。このような述語では、「を」が付いた名詞句は “飾る場所” であったり “飾る手段となる移動物” であったりする。そして、そのどちらであるかは他の助詞「で」または「に」によって規定される。

それでは、「太郎がテーブルを飾った」や「太郎が花を飾った」のように、「で」や「に」がない場合はどうなるであろうか。我々には「テーブル」や「花」という単語の意味から、「を」が付いた名詞句が “飾る場所” を表すのか、“飾る手段となる移動物” を表すのかは明らかなことが多い。

しかし、「を」が付く名詞によっては、解釈があいまいになる場合がある。たとえば、「太郎が車を飾った」の「車」は、“飾る場所” であるのか、それとも、車自体が “飾る手段となる移動物” を表すのか、どちらの解釈も可能である。

この研究では、このような状況において、我々人間はどのような情報を用い、どのような処理を行なうことにより、述語の意味フレームを決定しているかをモデル化し、そのプログラムを作成し、シミュレーションを行なう。

3. 意味フレームの決定方法

(1) 辞書の定義

まず、述語の語彙記述として、次のような、統語フレーム、意味フレーム、主題役割の属性表示を用いる。

飾る
- s⟨Subj, Obj⟩
- m⟨Agent, Theme, Location⟩
- Agent: activity 0.3, susceptivity 0.7
- Theme: change-of-location 0.3, object-of-display 0.7
- Location: space 0.3, result-of-change 0.7

1 行目の s⟨Subj, Obj⟩ は、述語「飾る」の統語フレーム (syntactic frame) が主語 (Subj) と目的語 (Obj) から構成されていることを示す。2 行目の m⟨Agent, Theme, Location⟩ は、「飾る」の意味フレーム (semantic frame) が Agent(動作主)、Theme(移動物)、Location(場所) から構成されていることを示す。次項 (2) で述べる方法は、どの統語フレームの名詞句が意味フレームのどのスロットを埋めるかを決定するものである。

さらに、この語彙記述では、Agent や Theme や Location というような意味役割に貢献する属性を述語ごとに度合い付きで指定する。本研究では、このような属性を具体的な属性ではなく、人間が自然言語を解析する際に用いているであろう抽象的な属性で記述する。このような単語の属性を我々は「認知的属性」と呼ぶ。「飾る」の場合、Agent 性に対する activity(主体性) という属性が貢献する度合いが 0.3 で、susceptivity (感性) という属性が貢献する度合いが 0.7 である。同様に、Theme 性に対する change-of-location (移動性) という属性が貢献する度合いが 0.3 で、object-of-display (展示対象性) という属性が貢献する度合いが 0.7 であり、Location 性に対する space (空間性) という属性と result-of-change (変化の残影性) という属性の貢献の度合いが、それぞれ、0.3 と 0.7 であると考える。そして、このような述語の意味フレームの主題役割を決定するのに用いられる属性を特にその主題役割における “注目属性” という [7]。

名詞については、それが持っている属性と、その強さを語彙記述に含める。たとえば、「車」、「ロビー」の記述は次のようになる。

車
- change-of-location: *middle*
- concrete: *high*
- space: *middle*
- result-of-change: *middle*
- object-of-display: *low*

ロビー
- concrete: *high*
- space: *high*
- result-of-change: *high*

ここで、*high* や *middle* は各属性上のファジィ集合で、例えば、次のように定義する。

$$
\begin{aligned}
high &= \{0.222/7, 0.778/8, 1/9, 1/10\} \\
middle &= \{0.222/3, 0.778/4, \\
&\qquad 1/5, 0.778/6, 0.222/7\} \\
low &= \{1/0, 1/1, 0.778/2, 0.222/3\}
\end{aligned}
$$

ここでは、全体集合を $\{0, 1, 2, \ldots, 10\}$ としている。このとき、たとえば、「車」の change-of-location (移動性) が *middle* であるとは、「車」というクラ

スが “移動” という属性を持っており、その属性の「車」との結びつきが中くらいである、ということを意味する。それに対して、「ロビー」は現実世界では “移動” しないものであるので、「ロビー」の記述に change-of-location は含まれない (0 と考える)。concrete (具体性) は、「車」も「ロビー」も抽象的な概念でもないし、集合名詞でもないので、*high* と見なせる。space (空間性) は「ロビー」との結びつきが強い属性であり、*high* と考えられる。「車」にも空間性はあるが、少ししかないので、*middle* と言えよう。result-of-change は変化の影響を残しやすいかどうかの性質であり、空間性と同様、「ロビー」との結びつきは強いが、「車」との結びつきはそれほどではない。つまり、「ロビー」は、明るくなったり、暗くなったり、人が集まったり、調度品が変わったり、いろいろな変化が見られることが想定されるが、「車」については、人や荷物が中にあったりなかったり、ボディがへこんだり、ある程度の変化は想定できるが、それ程変化が期待されるものではない。

(2) 意味フレーム決定アルゴリズム

前項 (1) で述べたような考え方を用いて、2 節で述べたような文を処理する方法を提案する。具体例として、「車」と「ロビー」とがあって、これらを「飾る」ときに、 Theme が「車」で、 Location が「ロビー」である、と判断する過程を示す。ここでは簡単のため Agent に関する説明は省略する。

(1) 述語「飾る」に対する名詞句「車」と「ロビー」の Theme 性を計算する。
(2) 述語「飾る」に対する名詞句「車」と「ロビー」の Location 性を計算する。
(3) Theme 性と Location 性の合計値が最も大きくなるような名詞をそれぞれ Theme と Location とする。

ここで、(1) における計算について、具体的に述べよう。述語「飾る」の Theme を決定するのに用いられる属性 (注目属性) は、「飾る」の Theme を見ると、change-of-location 0.3、object-of-display 0.7 であることが分かる。それらの名詞「車」における値は、辞書から change-of-location が *middle* で、object-of-display が *low* となる。したがって、「車」の Theme 性は重み付きの和を用いて、

$$\mathrm{Theme}(\text{車}) = \mathit{middle} \times 0.3 + \mathit{low} \times 0.7$$

となり、これを拡張原理で計算すると、

$$\begin{aligned}
&\mathrm{Theme}(\text{車}) = \\
&\quad \{0.222/3, 0.778/4, 1/5, 0.778/6, 0.222/7\} \times 0.3 \\
&\quad + \{1/0, 1/1, 0.778/2, 0.222/3\} \times 0.7 \\
&= \{0.222/0.9, 0.778/1.2, 1/1.5, \\
&\qquad\qquad 0.778/1.8, 0.222/2.1\} \\
&\quad + \{1/0, 1/0.7, 0.778/1.4, 0.222/2.1\} \\
&= \{0.222/0.9, 0.778/1.2, 1/1.5, 0.222/1.6, \\
&\quad 0.778/1.8, 0.778/1.9, 0.222/2.1, 1/2.2, \\
&\quad 0.222/2.3, 0.778/2.5, 0.778/2.6, 0.222/2.8, \\
&\quad 0.778/2.9, 0.222/3, 0.778/3.2, 0.222/3.3, \\
&\quad 0.222/3.5, 0.222/3.6, 0.222/3.9, 0.222/4.2\}
\end{aligned}$$

となる。小数点以下を四捨五入してまとめると、

$$\mathrm{Theme}(\text{車}) = \{0.778/1, 1/2, 0.778/3, 0.222/4\}$$

となる。同様に、名詞「ロビー」 では、change-of-location も object-of-display もないので、「車」の Theme 性は 0 となる。

さらに、(2) の Location 性は「飾る」の Location を見ると、space 0.3、result-of-change 0.7 となっている。名詞「車」は space が *middle* で、result-of-change も *middle* であるので、「車」の Location 性は重み付きの和を用いて、拡張原理で計算し、小数点以下を四捨五入してまとめると、

$$\begin{aligned}
\mathrm{Location}(\text{車}) = \{&0.222/3, 0.778/4, \\
&1/5, 0.778/6, 0.222/7\}
\end{aligned}$$

となる。また、名詞「ロビー」は、space も result-of-change も *high* なので、同様に、

$$\begin{aligned}
\mathrm{Location}(\text{ロビー}) = \{&0.222/7, 0.778/8, \\
&1/9, 1/10\}
\end{aligned}$$

となる。

以上により、「飾る」に対する「車」と「ロビー」の Theme 性と Location 性が求められた。そして、(3) でこれらの値の合計が最も大きくなる組み合わせを用いて、「車」と「ロビー」を意味フレームのスロットに割り当てる。このとき、一般的には、ファジィ集合同士の大小比較を行なう必要があるが、今のところは、ファジィ集合の重心を用いている。

以上の過程で、「車」と「ロビー」の統語フレームを用いていないことに注意する必要がある。このことは、統語解析のプログラムを簡単化できる

ことを意味する。また、統語上と意味上の矛盾などの評価にも利用できる。

(3) システムのインプリメント

前項 (1)、(2) で述べた手法を ファジィ・フレーム・システム [6] を用いて、インプリメントした。

まず、名詞の辞書を定義するためのファジィ集合を定義する。ファジィ集合の定義域は関数 defrange で定義し、ファジィ集合はファジィ・フレーム・システムの関数 $term を使って次のように定義する。以下、各種の定義用関数の値の表示は省略する。

```
(defrange 0 10)

($term
  (high {0.222/7, 0.778/8, 1/9, 1/10})
  (middle {0.222/3, 0.778/4,
             1/5, 0.778/6, 0.222/7})
  (low {1/0, 1/1, 0.778/2, 0.222/3}))
```

これで、[0, 10] 上のファジィ集合 *high*、*middle*、*low* が定義できた。

つぎに、ファジィ・フレーム・システムの関数 $def-class を用いて名詞を定義する。ここで、「ロビー」はクラス「部屋」の下位クラスとした。さらに、「人間」とその下位クラスとして「太郎」も定義する。

```
($def-class "車"
  (%super (%value "名詞"))
  (change-of-location (%value middle))
  (concrete (%value high))
  (space (%value middle))
  (result-of-change (%value middle))
  (object-of-display (%value low))
  )

($def-class "部屋"
  (%super (%value "名詞"))
  (concrete (%value high))
  (result-of-change (%value high))
  (space (%value high))
  )

($def-class "ロビー"
  (%super (%value "部屋"))
 )

($def-class "人間"
  (%super (%value "名詞"))
  (change-of-location (%value middle))
  (concrete (%value high))
  (activity (%value high))
  (susceptivity (%value high))
  )

($def-class "太郎"
  (%super (%value "人間"))
  )
```

次に、述語「飾る」を次のように定義する。ここでは Agent の定義も追加した。

```
($def-class "飾る"
   (%super (%value "動詞"))
   (%semantic-f
     (%value (agent theme location)))
   (%syntactic-f (%value (subj obj)))
   ("活用" (%super "ラ行５段"))
    (agent (%value
      ((activity 0.3)
       (susceptivity 0.7))))
    (theme (%value
      ((change-of-location 0.3)
       (object-of-display 0.7))))
    (location (%value
      ((space 0.3)
       (result-of-change 0.7))))
   )
```

このように意味フレームを%semantic-f スロットに、統語フレームを%syntactic-f スロットに記述する。

次に実行結果を示す。まず、述語に対する Theme 性や Location 性の計算は関数 semantic-frame-analyze が行う。

```
>(semantic-frame-analyze
              "飾る" "太郎" "車" "ロビー")
((agent (("太郎"
          {0.222/7, 0.778/8, 1/9, 1/10})
         ("車" 0) ("ロビー" 0)))
 (theme (("太郎"
          {0.778/1, 1/2})
         ("車"
          {0.778/1, 1/2,
                    0.778/3, 0.222/4})
         ("ロビー" 0)))
 (location
     (("太郎" 0)
      ("車"
       {0.222/3, 0.778/4,
                 1/5, 0.778/6, 0.222/7})
      ("ロビー"
       {0.222/7, 0.778/8, 1/9, 1/10}))))
```

このように semantic-frame-analyze は第一引数に述語名、第二引数以降に名詞を入れると、各述語の主題役割に対する適合度 (Theme 性などの値) を計算する。そして、実際に意味フレームのスロッ

トへ名詞を割り当てる関数が assign である。

```
>(assign "太郎" "ロビー" "飾る" "車")

("飾る" (agent ("太郎"))
        (theme ("車"))
        (location ("ロビー")))
0.66706090712742983
```

assign は内部で semantic-frame-anlyze を呼び出して、その結果から意味フレームの各スロットへの名詞の適合度の合計が最大になるような組合せを求めて出力する。意味フレームの下の数字は意味フレームの各スロットと割り当てられた単語との適合度の平均を定義域の上限で割って正規化したものである。この値は、出力された意味フレームが、意味的な整合性をもつ程高くなると言える。また、今回の方法では意味フレームの決定に統語情報を用いていないので、関数に入力として与える単語の順序を入れ換えても同じ結果になる。

他にも実行例をいくつか示す。まず、「花」と「花瓶」を次のように定義する。

```
($def-class "花"
  (%super (%value "名詞"))
  (change-of-location (%value high))
  (concrete (%value high))
  (object-of-display (%value high))
  )

($def-class "花瓶"
  (%super (%value "名詞"))
  (change-of-location (%value high))
  (concrete (%value high))
  (object-of-display (%value middle))
  (space (%value low))
  (result-of-change (%value middle))
  )
```

そして、「車」や「ロビー」にこれらを飾った場合や、「花」を「花瓶」に飾った場合をみてみる。

```
>(assign "ロビー" "花" "飾る")

("飾る" (agent nil)
        (theme ("花"))
        (location ("ロビー")))
0.89260000000000006

>(assign "車" "花瓶" "飾る")

("飾る" (agent nil)
        (theme ("花瓶"))
        (location ("車")))
0.55000000000000004

>(assign "花" "花瓶" "飾る")

("飾る" (agent nil)
        (theme ("花"))
        (location ("花瓶")))
0.6463000000000001

>(assign "太郎" "花瓶" "飾る")

("飾る" (agent ("太郎"))
        (theme ("花瓶"))
        (location nil))
0.74629999999999996
```

となり、各入力に統語情報 (格助詞) が含まれていないが、意味フレームを (意味的に) 正しく決定できている。

4. 直前の文から意味フレームを決定する方法

例えば「太郎が車を飾る」という文章を考える場合、「何か」を「車」に「飾る」場合と、「何か」に「車」を「飾る」場合の 2 通りの解釈が可能である。 省略された「何か」にあたる単語を補わないと正しい意味フレームが決定できない。しかし、「ロビーが殺風景なので」という文が直前にあった場合、「太郎が車を飾る」における「車」の主題役割は Theme であると決定できる。そして、「飾る」の Location となるのは直接「飾る」によって統語的に統制されていない「ロビー」が埋める。そこで、文献 [7] に基づいて、前の文の結果を利用できるようにした。すなわち、意味フレームの決定に際し、各主題役割の注目属性とスロットを埋めた単語を記憶しておき、次の意味フレーム決定時に空スロットがあった場合は記憶してある名詞を追加して意味フレームを決定する。まず、「太郎が車を飾る」のみで「飾る」の解析を行うと次の様になる。

```
>(assign "太郎" "車" "飾る")

("飾る" (agent ("太郎"))
        (theme nil)
        (location ("車")))
0.66706090712742983
```

ここでは「車」の主題役割は Location となる。

次に、「ロビーが殺風景だ」を解析した後、「太郎が車を飾る」を解析してみる。まず、述語「殺風景だ」と「美しい」を次のように定義する。

```
($def-class "殺風景だ"
  (%super (%value "形容動詞"))
  (%semantic-f (%value (location)))
  (%syntactic-f (%value (subj)))
  (location (%value
    ((space 0.2)
     (result-of-change 0.8))))
  )

($def-class "美しい"
  (%super (%value "形容詞"))
  (%semantic-f (%value (location)))
  (%syntactic-f (%value (subj)))
  (location (%value object-of-display))
  )
```

そして、「ロビーが殺風景だ」を解析すると次のようになる。

```
>(assign "ロビー" "殺風景だ")

("殺風景だ" (location ("ロビー")))
0.8926000000000006
```

となる。そして、「太郎が車を飾る」を解析すると次のようになる。

```
>(assign "太郎" "車" "飾る")

("飾る" (agent ("太郎"))
        (theme ("車"))
        (location ("ロビー")))
0.66706090712742983
```

このように、入力に"ロビー"がないにもかかわらず、出力に"ロビー"が現れている。

関数 assign が実行されると、変数*p-memory*に内容を記憶する。これは (assign "ロビー" "殺風景だ") を実行した直後の内容は次のようになっている。

```
>*p-memory*
{0.716/<result-of-change,"ロビー">,
                  0.183/<space,"ロビー">}
```

すなわち、意味フレーム決定時のスロットに対する名詞の適合度に、述語のスロットに記述されている重みをかけた値を正規化したもの[1]を記憶の度合いとして、注目属性と単語の対を記憶する。そして、関数 assign は意味フレームを決定の際、意味フレームに空スロットがある場合、意味フレームの各スロットの注目属性を用いて*p-memory*から名詞を検索、追加する。このとき、一度目の意味フレーム決定の際に埋まっているスロットに対しても、そのスロットを埋めている名詞より、高い適合度を示す名詞が*p-memory*に存在する場合はその名詞を追加する。そして、もとの入力に存在する名詞と、新たに追加した名詞を用いて再度意味フレームを決定する。

また、この記憶を保持する変数*p-memory*の中身は関数 assign を実行するたびに度合いが低下し、忘却されていく[2]。例えば、上の状態でさらに 2 回続けて「太郎が車を飾る」を解析してみる。

```
>(assign "太郎" "車" "飾る")

("飾る" (agent ("太郎"))
        (theme ("車"))
        (location ("ロビー")))
0.66706090712742983

>(assign "太郎" "車" "飾る")

("飾る" (agent ("太郎"))
        (theme nil)
        (location ("車")))
0.6963000000000014
```

このように 3 回目の解析では「ロビー」は現れない。これは「ロビーが殺風景だ」によって注目された認知的属性が忘却されたからである。

例をもう一つ示す。「花は美しい」「ロビーは殺風景だ」という文を解析した後、「太郎は飾った」という文の意味解析を行うと次のようになる。

```
>(assign "花" "美しい")

("美しい" (location ("花")))
0.8926000000000006

>(assign "ロビー" "殺風景だ")

("殺風景だ" (location ("ロビー")))
0.8926000000000006

>(assign "太郎" "飾る")

("飾る" (agent ("太郎"))
        (theme ("花"))
```

[1] 実際にはこれにシステムの変数*memory-power*をかける。初期値は 1.0 である。

[2] 実際のシステムでは度合いを 2 乗しているが、使用者が容易に変更できるようにしてある。

```
        (location ("ロビー")))
0.8926000000000006
```

すると、最後の文には Theme となる名詞も Location となる名詞も存在していないにもかかわらず、「飾る」に定義されたスロットはすべて埋められている。

また、assign は次のように文を「＋」で区切ることにより、「ロビーが殺風景なので」「太郎が車を飾った」というような文を続けて処理できる。assign は「＋」を読みこむたびに意味フレームを決定して出力する。実行例を示す。

```
>(assign "ロビー" "殺風景だ" +
         "太郎" "車" "飾る")

("殺風景だ" (location ("ロビー")))
0.8926000000000006

("飾る" (agent ("太郎"))
        (theme ("車"))
        (location ("ロビー")))
0.66706090712742983

>(assign "ロビー" "美しい" +
         "太郎" "車" "飾る")

("美しい" (location ("ロビー")))
0

("飾る" (agent ("太郎"))
        (theme nil)
        (location ("車")))
0.69630000000000014
```

となる。ここで注目してほしいのは「ロビーが殺風景なので、太郎が車を飾る」の「飾る」の Location には「ロビー」が埋まっているが、「ロビーが美しいので、太郎が車を飾る」の「飾る」には「ロビー」が出て来ていないことである。これは「美しい」の Location の注目属性は object-of-display であるが、「ロビー」には object-of-display が存在しないので、何も記憶されない為である。

つまり、「ロビーが殺風景なので、太郎が車を飾る」において、「ロビー」が「飾る」の Location となるのは、「ロビー」の「殺風景だ」における主題役割が Location でありからではなく、「殺風景だ」の Location における注目属性が result-of-change と space であり、それらの属性は、「飾る」の Location が注目する属性でもあるからである。このような、節と節、文と文を越えた述語–項関係を成立させるのに用いられる情報が主題役割そのものでなく、主題役割によって注目される認知的属性である。それを示す次の例を見てもらいたい。

```
>(assign "花" "美しい" +
         "太郎" "車" "飾る")

("美しい" (location ("花")))
0.8926000000000006

("飾る" (agent ("太郎"))
        (theme ("花"))
        (location ("車")))
0.76173333333333348
```

ここでは、「花」は「美しい」の Location と解析されたが、「美しい」の Location によって注目され、記憶された認知的属性が、「飾る」の Theme の注目する属性であった為に「飾る」の意味フレームに「花」が現れている。

5. 簡単な自然言語による入力

本システムには自然言語による入力から、格助詞による統語情報は考慮せずに、辞書に登録されている名詞と、述語 (動詞、形容詞、形容動詞) を抜きだして、関数 assign に渡す機能をもつ関数 assign-n もある。これによる実行例をいくつか示す。

```
>(assign-n
    "ロビーが殺風景なので、太郎が車を飾る。")

("殺風景な" (location ("ロビー")))
0.8926000000000006

("飾る" (agent ("太郎"))
        (theme ("車"))
        (location ("ロビー")))
0.66706090712742983

>(assign-n
   "花瓶が殺風景だ。花を飾る。")

("殺風景だ" (location ("花瓶")))
0.40000000000000008

("飾る" (agent nil)
        (theme ("花"))
        (location ("花瓶")))
0.6463000000000001

>(assign-n
   "花をロビーと車に太郎が飾る")
```

```
("飾る" (agent ("太郎"))
        (theme ("花"))
        (location ("ロビー" "車")))
0.7944499999999999

>(assign-n
   "花瓶が殺風景だ。車を飾る。")

("殺風景だ" (location ("花瓶")))
0.4000000000000008

("飾る" (agent nil)
        (theme nil)
        (location ("車")))
0.5000000000000011

>(assign-n
   "花瓶を車に太郎が飾る")

("飾る" (agent ("太郎"))
        (theme ("花瓶"))
        (location ("車")))
0.6642000000000001
```

また、本システムのその他の機能として、意味フレームが決定されたときに起動する関数を述語ごとに定義できる。その関数は関数%def-operateで定義できる。これを実行するには関数 assign の代わりに assign*を用いればよい。この機能を用いると、例えば意味フレームを決定するたびに辞書の定義を更新したり、文脈情報を構築してゆくなどの使い方が考えられる。

6. おわりに

以上、自然言語における名詞句の意味フレームをファジィ演算により決定する方法を提案した。さらにその方法を用いて、簡単な文ならば統語情報を用いずに意味フレームを決定できるシステムをファジィ･フレーム･システムを用いて作成した。しかし、検討すべき興味深い課題が多く残っている。

例えば、現在は意味役割への貢献度の計算に用いる認知的な属性により名詞を定義しているが、名詞レベルではもう少し具体的な属性で定義しておいて、その間の変換をファジィルールなどで行ないたい。また、現在は統語情報を意味フレーム決定に全く用いないが、これでは複雑な構造の文は処理出来ないため、統語情報もある程度扱えるようにしたい。また逆に、統語解析に今回提案した認知的属性とファジィ理論を用いた方法を使うことにより、意味的な整合性のある統語解析を行うプログラムを作成できるのではないかとも考えられる。

参 考 文 献

[1] L.A.Zadeh (1978) : "PRUF—A Meaning Representation Language for Natural Languages,", *International Journal of Man-Machine Studies*, Vol.10, pp.395–460

[2] L.A.Zadeh (1986) : "Test-Score Semantics as a Basis for a Computational Approach to the representation of Meaning ," *Literary and Linguistic Computing*, Vol.1, pp.24–35

[3] 馬野 (1993) :「ファジィ集合に基づく自然言語の意味表現」、情報処理、Vol.34、No.1、pp.21–28

[4] 馬野 (1989) :「ファジィ集合の概念を用いた日本語質問応答システムの拡張—ファジィ推論の導入と一致度の処理方法の追加—」、第 5 回ファジィ システム シンポジウム、pp.433–438

[5] 藤本、菅野 (1997) :「ファジィ言語処理システムに関する一考察」、第 13 回ファジィ システム シンポジウム、pp.289–292

[6] 角井、馬野、宮原 (1989) :「ファジィ理論に基づくフレーム・システムの実現」、情報処理学会 第 39 回 (平成元年後期) 全国大会、pp.70–71

[7] 船山、野田、馬野、宇野、中村、佐藤 (1998) :「節境界を越えた述語—項関係の計算モデル」、言語処理学会 第 4 回年次大会発表論文集、pp.262–265.

[8] 馬野、船山、宇野、野田、佐藤、中村 (1998) :「認知要素とファジィ理論を用いた自然言語理解システム—意味フレームの決定—」、第 14 回ファジィシステムシンポジウム、pp.331–334

[9] 馬野、後藤、船山、宇野、野田、佐藤、中村 (1998) :「認知要素とファジィ理論を用いた意味フレーム決定システム—ファジィ・フレーム・システムによる実現と機能拡張—」、第 15 回ファジィシステムシンポジウム、pp.241–244

[問い合わせ先]
〒599–8531
堺市学園町 1 – 1
大阪府立大学 総合科学部 数理・情報科学科
馬野 元秀
TEL: 0722–54–9675
FAX: 0722–54–9930
Internet: umano@mi.cias.osakafu-u.ac.jp