談話理解システム作成のための基礎的研究 — Semantic Matcher の実現

田中穂積
（東京工業大学工学部情報工学科）

1．はじめに

複数の文の連鎖からなる談話は、前文に含まれる情報（登場人物、出来事、状態等）を後続文で参照しつつ、新しい情報を付加して行く形で進行する。談話理解システムには、この様な現象を扱うことができるモデルを組み込まなければならない。

まず前文に含まれる情報の後続文からの参照は、文を読み進む過程で、文に含まれる登場人物、出来事、状態などが我々の記憶の中に残され、それらが後続文で参照されることであると考えられる。ここで、登場人物、出来事、状態などは、何等かの実体として我々の記憶の中に留められることになる。この様な実体のことを以下ではオブジェクトと呼ぶことにする。次に新しい情報の付加とは、前文にはなかった新しいオブジェクトや、オブジェクト間の関係が記憶の中に新たに作り出されることである、と考えることができる。

以上のことから、談話理解のモデルとして重要なものの一つに、記憶のモデルがあることが分かる。そしてこのモデルは、談話理解が進むにつれて、記憶の内部でオブジェクトの生成を伴う動的なモデルである。それによれば、言語学で前方照応と呼ぶ現象は次の様に説明される。一つの文を読み進む過程で、様々なオブジェクトが我々の記憶の中に作り出される。前方照応とは、新しく作り出されたオブジェクトの幾つかが、すでに前文で作り出されたオブジェクトと同一であるということを認識することである。別の言い方をすれば、記憶内部に生成されるオブジェクト相互の同一性を認識することによって、前方照応の問題を解くことができる。前方照応の問題は、談話理解の基礎であるから、オブジェクト相互の同一性を認識する問題は、談話理解システムを作成する場合の最も基礎的な問題であると言えよう。

オブジェクトの同一性の認識は、談話の中で使われる言葉の同一性を手がかりにすれば容易であると思われるかもしれない。第２章では、オブジェクトの同一性を認識することが、それほど容易な問題ではないことを実例により説

明する。

第３章では、計算機の中でオブジェクトをどの様な形式で表現するかを述べる。これは知識表現形式の問題である。我々はＤＣＫＲと呼ばれる形式でオブジェクトを表現する。これはロジックプログラミングの枠内での知識表現の形式である。ＤＣＫＲ形式のオブジェクト表現にはどの様な特徴があるかをも論じる。

第４章では、オブジェクトの同一性を認識するアルゴリズムについて述べる。その一部は、人工知能の研究者の間で 、semantic matcher [Nilsson 80], forced match [Bobrow 77]と呼ばれているものとして一般化できる。このアルゴリズムはＤＣＫＲによる知識表現形式を用いて比較的簡単に実現することができる。

２。　前方照応とオブジェクトの同一性の認識

前方照応と記憶内に生成されるオブジェクト相互の同一性を認識する問題を理解するために、以下の例文を考えてみよう。

（１）　机の上に赤い箱がある。箱の左に鉛筆がある。
（２）　目と目が合った。
（３）　胃に入った食べ物は胃の底にたまる。
（４）　公園で鳩が一羽うずくまっていた。その鳥は足に怪我をしていた。
（５）　自転車を売った。その金で本を買った。
（６）　自転車を買った。サドルの高さが低すぎたので調節した。
（７）　塩素酸ナトリウムを熱し、発生する気体を水上置換で集気ビンに集める。それは水に溶けるだろうか？

（１）の例文について、後続文に現われる箱は、前文の赤い箱であると容易に認識できる。どちらも箱という（同じ）単語が使われているからである。したがって、前文の赤い箱に対応する（記憶内の）オブジェクトと、後続文で作られる箱のオブジェクトとが同一であることを認識するアルゴリズムを考えることは、単語の同一性を見れば良いわけだから、自明の事のように思われる。

果たしてそうだろうか。

まず、文（２）と（３）とを見ると、問題はそう単純ではないことに気付く。（２）も（３）も一文中に目と置という単語がそれぞれ２つづつ現われる。（２）の文中に現われる２つの目を単語が同じであるという理由で、同一のオブジェクトであると認識してはならない。相異なる２つのオブジェクトを記憶の中に作らなければならない。ところが（３）の２つの置と言う単語は同一の置を指している。したがってこの場合には、２つの異なるオブジェクトを記憶の中に作り出してはならない。

（４）、（５）、（６）、（７）の例では後続文で下線を引いた単語に対応するオブジェクトと照応するものが、前文中では同一名の単語として現われていない。（４）では、後続文中のその鳥というオブジェクトに照応する前文中のオブジェクトは鳩である。この場合には、我々は「鳩は鳥の一種である」という常識を働かせた推論を行い、オブジェクト相互の同一性を認識していると思われる。

一方、（５）、（６）の例では、前文中に直接現われないオブジェクトを後続文中で参照している。（５）では、前文の理解の結果として「自転車を売ればお金が手に入る」という常識を用いた推論結果を後続文のその金が参照している。（６）では、前文で自転車と言うオブジェクトが記憶内に生成されるとともに、「自転車にはサドルやハンドル等が付いている」という常識を用いた推論結果を後続文のサドルが参照している。

（５）、（６）では、前文を意味理解する時に推論を十分に行ない、推論結果を記憶内に蓄えておくことができれば、後続文に現われるオブジェクトの意味が比較的はっきりしているために、それから得られる情報を用いて、それと照応するものを探し出すことは、比較的容易である。ところが（７）では、後続文に現われるそれはの意味が中立的で不定に近く、ほとんど全てのオブジェクトを指示し得るので問題がある。これについては、最後の章で簡単に説明する。

（１）から（４）に示した例については、４章で述べるアルゴリズムを直接応用して、照応関係を決めることができる。（５）と（６）の例については、前文の意味理解時に十分推論を行うことができれば、（１）から（４）の例と同様に、４章のアルゴリズムによって直接照応関係を決めることができる。

３。 DCKRによる知識表現

第４章で説明するオブジェクトの同一性を認識するアルゴリズムで用いる知識表現形式ＤＣＫＲを説明する[小山 85],[田中 86]。

述語論理式で表現した知識と、フレームまたは オブジェクト で表現した知識との間の関係は [Hayes 80],[Nilsson 80],[Goebel 85], [Bowen 85]等により明らかにされている。[Nilsson 80]は、述語論理式で表現した知識を一度オブジェクト形式に変換し,それを意味ネットワーク形式で表現する方法をのべている。それによれば、 さらに推論は、意味ネットワークをたどる操作に還元される。したがって、推論／問題解決をおこなうためには、別途この意味ネットワークに対する特別なプログラム（インタープリタ）を作る必要がある。

それに対して筆者らはDCKR(Definite Clause Knowledge Representation)とよぶ知識表現形式を開発している [小山 85] 。DCKRはホーン節形式をベースにした知識表現形式であり、意味ネットワーク形式の知識をDCKRで容易に記述することができる。DCKRで記述した知識に関する推論を行なうプログラムのほぼ全ては、Prologに組み込みの機能で代用することができる。推論を行なうための特別なプログラムを作る必要はない。

３・１。 ＤＣＫＲによるオブジェクトの表現

ＤＣＫＲによるオブジェクトの表現を実例を用いて説明する。

```
:-op(100,yfx,~),
  op(100,yfx,:),
  op(90,xfy,#).

01) sem(human,P,S) :-
        isa(mammal,P,[human|S]).
02) sem(elephant,P,S) :-
        isa(mammal,P,[elephant|S]).
03) sem(mammal,bloodTemp:warm,_).
04) sem(mammal,P,S) :-
        isa(animal,P,[mammal|S]).
```

```
05) sem(animal,P,S) :-
        hasa(body,P,[animal|S]).
06) sem(animal,age:X,S) :-
        bottomof(S,B),
        sem(B,birthYear:Y,_),
        X is 1985 - Y.
07) sem(clyde#1,age:5,_).
08) sem(clyde#1,P,S) :-
        isa(elephant,P,[clyde#1|S]).
09) sem(elephant#1,birthYear:1980,_).
10) sem(elephant#1,P,S) :-
        isa(elephant,P,[elephant#1|S]).
11) sem(elephant#2,birthYear:1982,_).
12) sem(elephant#2,P,S) :-
        isa(elephant,P,[elephant#2|S]).
13) sem(mcCarthy#1,address:stanford,_).
14) sem(mcCarthy#1,nationality:american,_).
15) sem(mcCarthy#1,
              affiliation:stanfordUniversity,_).
16) sem(mcCarthy#1,P,S) :-
        isa(human,P,[mcCarthy#1|S]).
17) sem(mister5G#1,address:japan,_).
18) sem(mister5G#1,P,S) :-
        isa(human,P,[mister5G#1|S]).
19) sem(misterAI#1,address:america,_).
20) sem(misterAI#1,P,S) :-
        isa(human,P,[misterAI#1|S]).
21) sem(america,capital:newYork,_).
22) sem(america,climate:temperate,_).
23) sem(america,P,S) :-
        T = [america|S],
        isa(coutry,P,T);
        hasa(california,P,T);
```

```
        ..............  .
24) sem(california,P,_) :-
        T = [californiaIS],
        isa(state,P,T);
        hasa(stanford,P,T);
        ..............  .
```

ここでDCKRの記述で重要になる sem述語、isa 述語、hasa述語の意味を、上記したDCKRによる記述例にしたがって説明する。

sem 述語の第一引数は、オブジェクト名である。オブジェクトには大別して個体とプロトタイプがある。 心理学者は、 しばしばプロトタイプのことをstereotypeとよぶことがある。オブジェクト名のうち#の付いたものは個体名を、また#の付かないものはプロトタイプ名を表わす。たとえば07),08) に現われるclyde は個体名であり elephantはプロトタイプ名である。個体名が等しいsem 述語をヘッドとするホーン節の集合は、一つの個体を表わす。たとえば、07)-08) は、clyde という個体を表わす。プロトタイプ名が等しいsem 述語をヘッドとするホーン節の集合は、一つのプロトタイプを表わす。たとえば、03)-04) はmammalというプロトタイプを表わす。 ここで（ホーン節形式の）DCKRによるオブジェクトの表現は、完全にコンパイルすることができる事に注意しよう。知識のコンパイルは高速化につながるので、この事はDCKRによる知識表現の好ましい性質であるといえる。

sem 述語の第二引数は、スロット名とスロット値の対である。 たとえば01) の記述は、個体clyde の age が 5 である、という事実を表わす。そしてage がスロット名で、5 がスロット値である。スロット名とスロット値の対を以下ではＳＶ対とよぶ。したがって07) の age:5 はＳＶ対である。

08) の記述は、clyde がelephant であると読む。ここで08) は上位からの知識の継承（inheritance of knowledge）をそのまま記述したものであることに注意されたい。 08)は、elephant が性質 P を持てばclyde もその性質 P を持つと読むのである。01),04), はhumanがmammal であり、mammalがanimalであるという事実を記述している。後述するように知識の継承はPrologに組み込みのユニフィケーションによって、自動的になされることにも注意されたい。さて、05) はanimalがbodyを持っている、という事実を記述している。知識の継承で用いる述語isa は、オブジェクトの階層をたどるための述語である。

isa の定義をhasaの定義とともに以下に示す。

```
25) isa(Upper,P,S) :-
        P = isa:Upper;
        sem(Upper,P,S).
26) hasa(Part,X:Y,S) :-
        X == hasa,
        (Y = Part;
         sem(Part,hasa:Y,S)).
```

３・２。　ＤＣＫＲによる種々の推論

ＤＣＫＲ形式の知識が与えられると、Prolog に組み込みの推論機構を用いて種々の推論を行うことができる。これを実例により説明する。

```
?-sem(elephant#1,P,_).
```

を実行すると

```
P = birthYear:1980;
P = isa:elephant;
P = isa:mammal;
P = bloodTemp:warm;
P = isa:animal;
P = age:5;
P = hasa:body;
```

のようにelephant#1に関する知識を自動的に次々に得ることができる。知識継承によりelephant#1の上位にある知識（ＳＶ対；性質）が全て得られていることに注意されたい。elephant#1 には age の記述がないが、P = age:5 が得られている。これについては後述する。

さらに

```
?-sem(X,Y,_).
```

を実行すれば、ＤＣＫＲで表現されている全ての知識を、X と Yの対として得ることができる。

また、Prologの特徴を生かして、

```
?- sem(X,isa:mammal,_).
```

を実行すれば、下位の個体またはプロトタイプを上位のmammalから、アクセスすることができる。だだし、このゴールは、知識継承を行なうホーン節の全てのヘッドとユニファイ可能になり、その多くは最終的に失敗することになるから、知識継承を行なうホーン節の数が増えれば増えるほど計算コストがかかる。

最後に述語hasaの機能を調べるために、読者は

```
?- sem(america,hasa:X,_).
```

を実行して見てほしい。

ここで elephant#1 には age に関する記述がないにもかかわらず、最初の例では P = age:5 が得られていた。その理由を説明する。

01) の記述を別の観点から眺めると、01) は、個体の世界からプロトタイプの世界を呼び出し、プロトタイプが持っている情報を引き出すための記述であると見なすことができる。DCKRでは、isa 述語により一旦プロトタイプの世界に入れば、プロトタイプの世界に存在する全てのプロトタイプにアクセスすることができる。しかし、個体は動的に作られるものであるから、あらかじめ個体の世界の様子を知ることはできない。DCKRでは、プロトタイプの世界から個体の世界の様子を知る一つの手段として、06) のボディーで使われている述語bottomofが用意されている。

```
06.1) bottomof([Result|Y],Result) :-
              (var(Y);atomic(Y)),!,nonvar(Result).
06.2) bottomof([X|Y],Result) :-
              bottomof(Y,Result).
```

sem 述語の第三引数には、階層をたどった道筋がスタックされる。これをbottomof述語の第一引数に与えて、プロトタイプの呼び出し元（プロトタイプの世界を呼び出した個体）が何であるかを知ることができる。そして、プロトタイプの世界から、その個体が持っている知識を引き出す事ができる。たとえば 06) は、bottomof述語を用いて呼び出し元 B を知り、B のbirthYear を用いてage を計算する知識である。それにより、

```
?-sem(elephant#1,age:X,_).
```

を実行すると、isa 述語の働きによって10)から02),03),04),05)を経て06) に至り 06) のボディを実行して

```
X = 5
```

を得ることができる。06）のボディの記述は、ＤＣＫＲによる手続き付加の例になっている。しかもこの手続きは知識継承により、下位のオブジェクトからアクセスされている。

これまでの説明であらわれた推論は、Prologに組み込まれたインタープリタによって自動的になされる。DCKRより更に高級な知識表現形式ＳＲＬ／０が開発されている。ＳＲＬ／０で記述した知識は変換されてＤＣＫＲ形式になる。これについては［田中 86］、［奥村 86］を参照されたい。

４。　オブジェクトの同一性を認識するアルゴリズム

４・１　前方照応の処理

前方照応の処理は、オブジェクト相互間の同一性を認識する問題に帰着されるが、その認識は必ずしも容易であるとはいえないことを２章で説明した。特に２章の例（２）以降で、どの様な問題があるかを明らかにした。前方照応を扱うための前提になる「オブジェクト相互間の同一性の認識」ということの意味は、「比較すべき２つのオブジェクトが同一であるとみなしても矛盾がない」という意味であることに注意しよう。

本稿では、以下に示すオブジェクトの同一性を認識するアルゴリズムを提案する。ただし後続文から前方照応するオブジェクト（先行詞）は、記憶に必ず存在するものとする。

ここで先行詞の候補となるオブジェクトを記憶から取り出し、それが o#1 であるとしよう。そして o#1 と照応すべきオブジェクトが o#2 であるとしよう。

［１］　o#1とo#2を作る時に使われた単語が同一ならば［２］へ、さもなければ［３］へ。

［２］　それらの単語が同一名詞句の中に含まれるなら、o#1 と o#2 とは異なるオブジェクトとしてリターン、さもなければ o#1 と o#2 とは同一であるとしてリターン。

［３］　４・３で説明する Semantic Matcher を呼ぶ。

ここで［２］は２章の例（２）と（３）を正しく処理するためのものであるが、これは筆者の推測（conjecture）であり、今後の検討を要する。意味情報ではなく構文情報を用いていることに注意されたい。

４・２。 Semantic Matcherの基本的な考え

人工知能の分野でも オブジェクト 相互間のパターンマッチを行なう必要がしばしば生じる。この方向を推し進めて、オブジェクト相互間のユニフィケーション（同一化）を基本計算機構に取り入れた言語 CILが開発されている [Mukai 85]。

オブジェクト 相互間の同一化を行なう場合の一つの問題は、オブジェクトを構成するスロットの並びに関する順序的な制約がないため、スロットの並びの順番に依存しない同一化が必要になることである。そのため、オブジェクトを一つの大きなデータ構造（たとえばリスト構造）として表現した時には、計算コストの問題が生じる。ところがDCKRを用いれば、スロットが一つのホーン節として表現されるため、同一化時に必要になるスロットの選択を、Prologに組み込みのバックトラック機能に任せることができるので、問題は若干軽減する。それでも、一般にオブジェクト相互間の同一化を基本計算機構に組み入れた言語を設計する場合には、CIL のように同一化は、意味ではなくシンタクスのレベルに留めざるをえないだろう。

一方、意味を考慮したオブジェクト相互間の同一性の認識と同一化を行なうことの重要性は、 [Bobrow 77]は Forced Matching として、[Nilsson 80]は Semantic Matching として、またColmeraureはPrologのユニフィケーションを拡張する試みとして論じている。以下では、２章で述べた例を用いて、オブジェクト相互の同一化を、DCKRをベースにして行なう方法を述べる。この時、DCKRの形式で記述された知識の体系に関する知識が必要にある。そのため次節で説明するアルゴリズムでは、以下のメタ知識を使う。

```
27) metakb(mammal,ntype:exclusive).
28) metakb(age,ltype:exclusive).
29) metakb(address,ltype:exclusive).
```

27) は、プロトタイプ mammal の直下にあるプロトタイプ相互（たとえばhuman とelephant）が両立しないことを表している。28) は ageのスロット値が、また29) はaddress のスロット値が異なれば、そのようなスロット相互は両立しないことを表している。 例をあげれば 27) は human である個体とelephantである個体とは同一化できないということ、また28) は、 ageが44才

の個体と55才の個体とは同一化できないことを表している。

ここでSemantic Matching の例を幾つか挙げる。我々は、２章のDCKRの記述から容易に次のi)-iv)のような推論を行なうことができる。

i) mcCarthy#1のaddress はstanfordで、misterAI#1のaddress はamerica であることが分かる。我々は、stanfordがamerica のなかにあるという事実を知っているから、misterAI#1とmcCarthy#1とを同一視し、同一化してもなんら矛盾はないと推論することができる。
ii) 同様にして、america とjapan は異なる国であるから、mister5G#1とmcCarthy#1とを同一化できないと推論することができる。
iii)現在の年号から生まれた年の年号を引いたものが年齢であるとすれば、年齢が5 才のclyde#1 と生まれた年が1980年の elephant#1 とを同一化しても良いと推論することができる（現在が1985年だと仮定する）。
iv) 同様な推論を行なって、clyde#1 とelephant#2とを同一化することはできないと結論することができる。
v) mcCarthy#1とclyde#1 とを同一化できない。なぜなら、前者は人間(human)であり後者は象(elephant)だからである。

このように、意味を知り二つのオブジェクトの同一化を行なう操作をSemantic Matching (Forced Matching)とよび、Semantic Matching を行なうプログラムをSemantic Matcherとよぶ。

４・３ Semantic Matcher

これまで、このようなSemantic Matcherの必要性はしばしば論じられてきたが、Semantic Matcherを作成したという試みはさほど多くない。それは、上記した程度のSemantic Matcherでさえ、相当複雑なものになると考えられるからである。しかし、DCKRによる知識表現を用いれば、その作成は比較的容易である。完全なものとはいえないが、我々は次の[A] から [F]に示すアルゴリズムによって、i)からv)に示した程度の同一化能力を持つSemantic Matcherを作ることができる。

[A] ： 同一化の対象になる二つの個体 o#1, o#2 に共通する上位の オブジ

エクト（Oi）があればそれを取り出し[B] へ、さもなければ[D] へ。
[B] ： metakb(Oi,ntype:exclusive)が成り立つ（Oiの直接１レベル下位のオブジェクトが互いにexclusive な関係にある）ならば[C] へ、さもなければ[A] へ。
[C] ： Oiの直接１レベル下位に相異なる二つの オブジェクト Oj,Ok があり、個体 o#1とo#2 の上位にそれぞれOjとOkが位置しているなら、同一化は失敗であるとしてリターン、さもなければ[A] へ。
[D] ： o#1 にＳＶ対 Ax:Bx が、 o#2にＳＶ対 Ax:By があれば、以下（次ページ）に示す集合Ｓを作り[E] へ（スロット名が共通なことに注意）、さもなければ o#1と o#2とを同一化してはならないという積極的な理由がないので、同一化が可能であるとして [F]へ。

```
S = {(Ax:Bx, Ax:By) | metakb(Ax,ltype:exclusive) &
                      (Bx == By  or
                       sem(Bx,isa:By,_)  or
                       sem(By,isa:Bx,_)  or
                       sem(Bx,hasa:By,_)  or
                       sem(By,hasa:Bx,_))  }
```

[E] ： Ｓが空集合でなければ同一化が可能であるとして[F] へ、空集合なら同一化は失敗であるとしてリターン。
[F] ： o#1 とo#2 とを等しいとおくために（同一化するために）、次の事実をassertする。

```
sem(o#1,P,S) :-
    isa(o#2,P,[o#1|S]).
sem(o#2,O,S) :-
    isa(o#1,O,[o#2|S]).
```

以上によりo#1 にはo#2 の、またo#2 にはo#1 の性質が自動的にうけつがれ o#1 とo#2 とは同一化される。

[A] から[F] のアルゴリズムは、以下に示すmkeq述語として与えられる。ただしmkeq述語の第三引数が "?"マークなら、同一化可能性のみを調べ、[F] のassertは行なわない。

```
mkeq(X,Y) :- mkeq(X,Y,_),!,abolish(unknownisa,2).
mkeq(X,Y) :- abolish(unknownisa,2),!,fail.

mkeq(X,X,_) :- !.
mkeq(X,Y,_) :-
        sem(X,isa:Y,_),
        sem(Y,isa:X,_),!.
mkeq(X,Y,_) :-
        % checks exclusive relations in isa relations.
        indiv(X),indiv(Y),
        % get a common node Z which is marked as 'exclusive.'
        sem(X,isa:Z,_),
        metakb(Z,ntype:exclusive),
        sem(Y,isa:Z,_),
        % get a node A and B that is one level lower than Z.
        sem(A,isa:Z,1),
        not(indiv(A)),
        sem(X,isa:A,_),
        sem(B,isa:Z,1),
        not(indiv(B)),
        A ¥== B,
        sem(Y,isa:B,_),
        !,fail.
mkeq(X,Y,_) :-
        indiv(X),indiv(Y),
        sem(X,Ax:Bx,_),
        Ax ¥== isa, %isa relations have already checked above.
        eqsem(Ax:Bx,Y,State),
        ((State==fail,!,fail);
        fail).
mkeq(X,Y,Mk) :-
        indiv(X),indiv(Y),
        mklink(X,Y,Mk),
```

```
        mklink(Y,X,Mk).
eqsem(Ax:Bx,Y,State) :-
        metakb(Ax,itype:exclusive),
        sem(Y,Ax:By,_),
        (Bx == By;
          % (Bx==dog amd By==pluto)
         sem(Bx,isa:By,_);
         sem(By,isa:Bx,_);
          % (By==tokyo and By==Japan)
         sem(Bx,hasa:By,_);
         sem(By,hasa:Bx,_);
         State=fail),!.

mklink(_,_,MK) :-
        MK == ?,!
mkink(X,Y,_) :-
        assertz((sem(X,Prop,S) :-
                        isa(Y,Prop,[X|S]))).

indiv(X) :-
        nonvar(X),Y#_ = X,nonvar(Y).
```

先に述べた同一化のための操作 [F]のために、３章で述べたisa 述語の定義を修正する必要がある。操作[F] により、階層をたどる道筋がループするからである。これは、isa 述語の第３引数を用いて、ループのチェックを行う事ができる。

４・４ 実験結果

4.3 節のプログラムを用いて Semantic Matcher の簡単な実験を行なった。

```
(a) ?- mkeq(x#1,y#1).
(b) yes
```

```
(c) ?- mkeq(x#1,misterAI#1).
(d) yes
(e) ?- sem(y#1,P,_).
(f) P = isa:x#1;
(g) P = isa:misterAI#1;
(h) P = address:america;
(i) P = isa:human;
(j) P = isa:mammal;
(k) P = bloodTemp:warm;
(l) P = isa:animal;
(m) P = isa:creature;
(n) no
(o) ?- mkeq(mcCarthy#5,mister5G#1).
(p) no
(q) ?- mkeq(mcCarty#5,misterAI#1).
(r) yes
(s) ?- mkeq(mcCarty#5,clyde#1).
(t) no
(u) ?- mkeq(elephant#1,clyde#1).
(v) yes
(w) ?- mkeq(elephant#2,clyde#1).
(x) no
```

(a)では、x#1 とy#1 という２つのオブジェクトを作り出し、それらを等しいと置いている[D] 。(c) ではx#1 と misterAI#1 とを等しいと置いている[D] 。したがって(e) でy#1 のもつ性質を聞くと(f)ー(m) の応答に見るように、y#1 はmisterAI#1の性質を受け継いでいることが分かる[F] 。(o),(q),(u),(w) に対する応答は [E]によるが、それぞれi),ii),iii),iv) で説明したSemantic Matchingの例になっている。(s) に対する応答は [C]によるが、これは(v) で説明したSemantic Matching の例になっている。ここで、 (u)と(w) では
elephant#1とelephant#2のいずれにも、age についての記述がないにもかかわらず、正しい応答を示していることに注意したい。その理由は ３・２節で既

に説明した。

5。 おわりに

複数の文の連鎖からなる談話を理解するためには、前文を受けながら談話が進行する過程で様々なオブジェクトが発生し、それらのうちどれとどれが同一であるかを推論することが必要になる。本稿では筆者らが開発したＤＣＫＲと呼ばれるオブジェクトの表現形式を説明し、オブジェクトの同一性を認識するアルゴリズムを提案した。またそれが談話理解に重要な役割を果すことを説明した。部分的に不完全であるとは言え、２章の（１）から（６）に示した例を扱うことができる。（７）については、後続文中の「それ」が、前文中のいかなるオブジェクトをも指し得るため問題となる。この問題は、「文の焦点には目的格がくることが多い」という言語学的な事実により[Sidner 83]、前文から焦点を抽出しておくことにより、照応をとるべきオブジェクトの探索順を焦点を優先させて解決できると思われる。

本研究の延長上で筆者は、並立する名詞がどれとどれであるかを認識するアルゴリズムを考えて行きたいと思っている。そのためには、４章 で説明したオブジェクト間の同一性を判定するアルゴリズムではなく、二つのオブジェクト間の相似性を認識するアルゴリズムが必要になる。筆者は今後、Semantic Matcherを拡張して、こうした問題に応用することを考えている。 このようなSemanticMatcher の実現それ自身、長期を要する研究課題であるが、それは、人工知能の研究分野で今後重要になると思われるAnalogicalReasoning や、それをさらに一歩進めた学習の問題などにも応用可能だと思われる。オブジェクト相互の同一性、相似性を認識することが、談話理解、学習などといった困難な問題を解決するための第一歩であると考えている。

謝辞

本研究の契機は、ICOTの淵　一博所長とのディスカッションにある。筆者の良き指導者であり助言者である同氏に感謝する。筆者の所属する田中研究室の佐藤直人君は第三章で、池田光生君、上脇　正君、沼崎浩明君は第4.2 節で協力していただいた。斎藤佐知江さんには、本稿の作成で御苦労をいただいた。以上の諸氏に感謝する。

8．参考文献

[Bobrow 77] Bobrow,D.G. et.al.: An Overview of KRL-0, Cognitive Science, 1, 1, 3-46(1977).

[Bowen 85] Bowen,K.A.: Meta-Level Programming and Knowledge Representation, Syracuse Univ.,(1985).

[Goebel 85] Goebel, R.: Interpreting Descriptions in a Prolog-Based Knowledge Representation System, Proc. of IJCAI'85,711-716 (1985).

[Hayes 80] Hayes,P,J.: The Logic of Frame, in Frame Conceptions and Text Understanding, Walter de Gruyer, Berlin, 46-61(1980).

[小山 85] 小山晴生（他）：Definite Clause Knowledge Representation, Proc.of LPC'85, 95-106(1985).

[小山 86] 小山晴生：Prolog による structured object の表現形式と推論，東京工業大学理工学研究科修士論文,(1986).

[Mukai 85] Mukai,K.: Unification over Complex Indeterminates in Prolog, Proc.of LPC'85, 271-278(1985).

[Nilsson 80] Nilsson,N.J.: Principles of Artificial Intelligence, Tioga, (1980).

[田中 86] 田中穂積（他）：知識表現形式ＤＣＫＲとその応用、日本ソフトエア科学会論理と自然言語研究会資料、LNL86_6、1986.

[奥村 86] 奥村 学（他）：意味記述用言語SRL/0の設計とDCKR，情報処理学会自然言語処理研究会資料,54-5(1986).

[菅原 85] 菅原俊治：フレームシステムにおける継承機能の拡張，知識情報処理シンポジウム論文集,文部省科研費特定研究「多元知識情報」総括班，97-106(1985).

[Sidner 83] Sidner,C.: Focusing in the Comprehension of Definite Anaphora, in Computational Models of Discourse, The MIT Press 267-330(1983).