7Q-3

Vocalogの新たな語彙構成子

後藤　文太朗　　田中　譲

北海道大学　工学部

1　はじめに

自然言語において語彙は非常に強力な道具である。そして、計算機の上でも語彙は強力な道具と成り得る[1,2,3]。我々はPrologに語彙概念を導入したシステムVocalogを提案している[4]。Vocalogでは語彙として名詞、形容詞の二つを考えていた。語彙構成を行う操作には、名詞に対して論理演算、関数、形容詞、リスト化を考え、形容詞に対して論理演算、逆演算、合成演算を考えた。今回、我々はVocalogの表記法の変更を行い、語彙構成の新しい操作を導入した。これにより、統一的な語彙の取扱いが可能となり、語彙の表現力はさらに強化された。

2　Vocalogの新しい表記法

旧Vocalogではトップレベルにおいて述語を$p(x_1,x_2,\cdots,x_n)$と記述するかわりに、述語の引数の位置に対応する名詞$n_1,n_2,\cdots,n_n$を考え、$[n_1,n_2,\cdots,n_n]::(x_1,x_2,\cdots,x_n)$と表していた。これにより、知りたい概念を直接に表現することが可能となった。しかし、この形式はトップレベルに限られた特殊な形式であり、また、名詞と値が離れすぎていてわかりにくいという側面もあった。そこでトップレベルにおいては、“名詞を記述する”という、より一般的な表記法を導入する。そのために、次に述べる新たな語彙構成の操作をVocalogに導入する。

3　新たな語彙構成の操作

3.1　代入

名詞とその値の対を名詞とする新たな構成子として代入を導入する。代入は、

＜名詞＞　:=　＜値＞

のように記述する。この構成子の導入により先に述べたVocalogのトップレベルの表記法の一般化が成される。Prologにおいて$p(x_1,x_2,\cdots,x_n)$と記述される述語は$[n_1:=x_1,n_2:=x_2,\cdots,n_n:=x_n]$あるいは$[n_1,n_2,\cdots,n_n]:=[x_1,x_2,\cdots,x_n]$といった名詞で表現される。さらに、この構成子と他の構成子を組み合せて、制約の表現等が容易に行える。

例　(first_name:=taro+family_name)という名詞は、名詞first_nameの値として'taro'または名詞family_nameの取り得る値のみを取る。

3.2　射影演算

リスト化された名詞の一部を取りだす演算として射影演算を導入する。射影演算は次のように記述する。

＜名詞＞　＜＜　＜名詞＞

射影演算はリスト化して制約を課した名詞を取り出すのに利用できる。

例　family_nameが'goto'である人のfirst_nameのみを値として取ることができる名詞は、first_name＜＜[family_name:=goto,first_name]として表すことができる。

3.3　再帰演算

形容詞の再帰を表現する演算として再帰演算を導入する。再帰演算は次のように記述する。

＜形容詞＞**

例　祖先を表す形容詞ancestorを、親であるという形容詞parentを用いてparent**として定義することができる。

4　プログラミング例

Vocalogによるプログラミング例として文法規則の表現を取り上げる。DCGにおける以下の例を考える[5]。

```
sentence    -->  n_phrase,v_phrase.
n_phrase    -->  propernoun.
n_phrase    -->  posses,noun.
v_phrase    -->  v_intrans.
v_phrase    -->  v_trans,object.
object      -->  n_phrase.
propernoun  -->  [john].
posses      -->  [my].
noun        -->  [dog].
v_intrans   -->  [runs].
v_trans     -->  [likes].
```

Vocalogでこれらの文法規則を表現することにより、名詞としてsentence, component,period、形容詞としてsentence,n_phrase,v_phrase,object,propernoun,posses,noun,v_intrans,v_trans等を使えるようになる。これらの語彙の定義は後述する。これらの語彙を用いて色々な表現を行うことができる。

New vocabulary constructors in Vocalog

Fumitaro GOTO,　Yuzuru TANAKA

Hokkaido Univ.

例えば、この文法規則から導き出される文を名詞sentenceと代入を用いて、

Vocalog: sentence:=X.

として変数Xに求めることができる。また、'john likes my dog'という文がこの文法規則から導き出されるかどうかは、同様に名詞sentenceと代入を用いて、

Vocalog: sentence:=[john,likes,my,dog].

とすることにより確かめられる。そして、この文法規則から導き出される文の全てを求める場合、名詞sentence、代入、グループ・バイ演算を用いて、

Vocalog: sentence/[]:=X.

と表すことにより変数Xに求めることができる。さらに、文法規則を明記してそれにあった文を取り出すことも可能である。例えば、[Propernoun,V_intrans]となっている文は、形容詞としてpropernoun,v_intransと、名詞としてcomponent,period、合成演算、代入、射影演算を用いて、

Vocalog: (propernoun:v_intrans)@component
<<[(propernoun:v_intrans)@component,period]:=X.

と表すことにより、変数Xに求めることができる。

Vocalogによる具体的なプログラムを以下に示す。***Vocalog***でのプログラミングは、

Ⓐ基本となるPrologの述語を定義する(①)。
ⒷⒶの述語とⒸで用いる語彙とのインターフェース部分を定義する(②③④⑤)。
Ⓒ語彙構築を構成的に行い、その語彙を辞書に登録していく(⑥⑦)。

という三つの部分に大きく分けられる。***Vocalog***でプログラミングを行う利点は、プログラミングがⒸで行う"語彙を構成的に定義する"という容易な方法で行われるという点である。

①Prologの述語として次のものを考える。

```
d_list(X,Y) :- X==Y,!.
d_list([_|Y],Z) :- var(Y),Y=Z,!.
d_list([_|Y],Z) :- d_list(Y,Z).
d_list(X,X).

propernoun([john|X],X).
posses([my|X],X).
noun([dog|X],X).
v_intrans([runs|X],X).
v_trans([likes|X],X).
```

②基本属性を次のように定義する(第一引数:述語名,第二引数:述語の引数の位置に対応した属性のリスト)。

```
def_ba(d_list,[d_top,d_end]).
```

③基本修飾詞を次のように定義する(引数:基本修飾詞名,本体:定義)。

```
modifier(d(X)):-
        X(d(X)@d_top,d(X)@d_end),
        d(X)@d_end=d_top.
```

④基本名詞を次のように定義する(第一引数が基本名詞名で、第二引数がその構成を示す)。

```
dicNL0(component,d_top).
dicNL0(period,d_end:=[]).
```

⑤基本形容詞を次のように定義する(第一引数が基本形容詞名で、第二引数がその構成を示す)。

```
dicAL0(propernoun,d(propernoun)).
dicAL0(posses,d(posses)).
dicAL0(noun,d(noun)).
dicAL0(v_intrans,d(v_intrans)).
dicAL0(v_trans,d(v_trans)).
```

⑥形容詞を次のように定義する(第一引数が形容詞名で、第二引数がその構成を示す)。

```
dicAL(n_phrase,propernoun+posses:noun).
dicAL(object,n_phrase).
dicAL(v_phrase,v_intrans+v_trans:object).
dicAL(sentence,n_phrase:v_phrase).
```

⑦名詞を次のように定義する(第一引数が名詞名で、第二引数がその構成を示す)。

```
dicNL(sentence,sentence@component
        <<[sentence@component,period]).
```

5 おわりに

Vocalogに新たな語彙構成の操作として、代入、射影演算、再帰演算を導入した。これにより***Vocalog***の表現力はさらに強化され、また、語彙の統一的な取扱いが可能となった。なお、今回発表した***Vocalog***の機能はDEC10-Prologに準拠したSymbolics Prolog上に実現されている。これからの課題として、語彙の充実、ユーザー・インターフェースの改善を検討していきたい。

参考文献

1)Tanaka,Y.,"Information Space Model," Proc.2nd Workshop on Formal Bases for Databases,Dec. 1979, Toulouse.
2)Tanaka,Y.,"Vocabulary Building for Database Queries," Lecture Notes in Computer Science, vol.147:RIMS Symposia on Software Science and Engineering, Springer-Verlag, pp.215-232,1983.
3)Tanaka,Y.,"Roles of a Vocabulary in Knowledge-Based Systems," IFIP WG 10.1 Workshop,Gotenba,1987.
4)後藤文太朗,田中譲,"***Vocalog***:語彙構築機能を導入したProlog," 情報処理学会全国大会講演論文集 pp629-630(1988.9)
5)後藤滋樹,"PROLOG入門-知識情報処理の序曲-," サイエンス社