4K1-OS-2-6

汎用の言語変換に向けたプログラミング戦略とプラットフォーム

The programming strategy and the platform for universal conversion between natural languages

中村正治
NAKAMURA, Masaharu

This document describes the strategy to programme a semi-automatic natural language conversion and its platform system.

1. はじめに

自然言語の自動翻訳は、インターネット上の情報のアクセスに伴い、多くの言語により表現された情報の理解のためにしばしば利用されるようになってきている。残念ながら、この技術は個別言語間での「翻訳」であり、なおかつ、品質がかなり低いという点から、自然言語の相互変換といえるものではない。

本論文では、人手による判断をプロセスとして組み込んだ半自動の、なおかつ文脈自由な、汎用の自然言語変換プログラム実現のための戦略を紹介する。また、その適用分野として、古典語・古代語に焦点を当てて、このプログラムのプラットフォームとなる、言語センス・インターフェイスを備えた、多言語ワード・プロセッサーの要件とその実装を紹介する。

2. 言語変換

2.1 定義

V : Verb Complex : 無定義語。あるいは V の集合。
N : Noun Complex : 無定義語。あるいは N の集合。
V, N を合わせて意味担体;Meaning Bearer と呼ぶ。
注意 自然言語における意味単体としては、ほかに形容詞、副詞などがあるが、ここではこれらも N:Noun Complex である。
L : Language : 無定義語。あるいは L の集合
S : Sentence Element : 1つの V と任意の数の N の組
s ∈ S : s=(v ∈ V, n1,n2,…∈ N)
M : Modification : 修飾
v1,v2 ∈ V, n ∈ N : m ∈ M : m:(n,v1)→v2
あるいは
n1,n2,n ∈ N : m ∈ M : m:(n,n1)→n2
あるいは
s1,s2 ∈ S, n ∈ N : m ∈ M : m:(n,s1)→s2
C: Case : 格
m : (n,x1)→x2, x1,x2∈ N ないし V ないし S
のとき、c=m(n) として、この c を case (格)という。
この時、「n は格 c を持つ」といい、(n,c) と表す。
E : Envelope:封入
e ∈ E : e : s → (n,c) : c ∈ C, n ∈ N
V internal attribute : V の内部属性
言語ごとに定義される V に内在する属性。
N internal attribute : N の内部属性
言語ごとに定義される N に内在する属性。
内部属性クラス、内部属性インスタンスが定義される。
V 属性クラスとして、法・時・相・態・敬などある。
N 属性クラスとして、性・数などがある。

連絡先：中村正治，iLPL，東京都台東区上野桜木，電話・Fax 03-585-9177，praetorius@attglobal.net

これらは、合わせて「属性と内部その値」と表現する場合がある。
R : representation : 表現
t を文字列とした時、r ∈R, l ∈L として、
t1 = r (v,l) , t2 = r(n,l) をそれぞれ、言語 l における v の表現, 言語 l における n の表現、と呼ぶ。
RF : Formal Representation : 様式表現
1) t を文字列とした時、r ∈R, l ∈L として、
v ∈ V の持つ内部属性とその値を a1(i1),a2(i2),…
とした時、rf(v) = (r (v,l), a1(i1),a2(i2),…)
= (t, a1(i1),a2(i2),…)
を V の内部属性の様式表現と呼ぶ。
このときの(a1(i1),a2(i2),…)を V の conjugation と呼ぶ。
2) n ∈ N の持つ内部属性とその値を a1(i1),a2(i2),…
とした時、rf(n) = (r (n,l), a1(i1),a2(i2),…)
= (t, a1(i1),a2(i2),…)
を N の内部属性の様式表現と呼ぶ。
このときの(a1(i1),a2(i2),…)を N の declension と呼ぶ。
3) n ∈ N が格 c を持つ場合、(n,c)として、
rf(n,c) = (r (n,l), c)
= (t, c)
を N とのその格の様式表現と呼ぶ。
RM : Materialized Representation : 実体表現
1) t を文字列とした時、r ∈R, l ∈L として、
v ∈ V の持つ内部属性とその値を a1(i1),a2(i2),…
とした時、rm(v) = (r (v,l), a1(i1),a2(i2),…)
= t
を V の内部属性の実体表現と呼ぶ。
このときの t を V の conjugated と呼ぶ。
2) n ∈ N の持つ内部属性とその値を a1(i1),a2(i2),…
とした時、rm(n) = (r (n,l), a1(i1),a2(i2),…)
= t
を N の内部属性の実体表現と呼ぶ。
このときの t を N の declined と呼ぶ。
3) n ∈ N が格 c を持つ場合、(n,c)として、
rm(n,c) = (r (n,l), c)
= t
を N とのその格の実体表現と呼ぶ。
Syntax Graph : 構文グラフ
任意の数からなる v1,v1,…, n1,n2,…,s1,s2,…の組 G に対して、すべての V,N,の要素 x に対して、ある V,N の要素 y が存在して、m:x → y となり、かつ，一つを除くすべての S の要素 s にたいして、ある N,V の要素 z が存在して、m : s → z となる場合、G を完美な構文グラフと呼ぶ。
M により写像される V,N の要素を持たない s を「文: sentence 」と呼ぶ。ここで定義した「文」は、定義から明らかなように、所謂「単文」「複文」を含み「重文」を含まない。

GM : Graph Materialization 構文グラフの実体化
t を文字列とした時、g ∈G, l ∈L として、
gm(g,l) = t
を構文グラフの実体化ないし線形化と呼ぶ。
GF : Graph Formalization 構文グラフへの様式化
t を文字列とした時、g ∈G, l ∈L として、
gf(t,l) =g
を構文グラフへの様式化と呼ぶ。

2.2 言語変換のモデル

言語変換のモデルは、次のサービスから構成されるプロセとして定義される。

1) GF-service
与えられた文字列 t を構文グラフ化する。
2) RF-service
構文グラフ中のすべての意味担体を様式表現とする。
3) CB-service
全ての意味担体の言語を変換する。
4) RM-service
構文グラフ中のすべての意味担体を実体表現とする。
5) GM-service
構文グラフを線形化する。

2.3 言語変換の実装要素。

0) 全体
a) 言語識別の可能なワードプロセッサー
1) GF-service
a) V,N,S,M をシンボルとして表現する機能
b) 自由に組み換えのためのグラフィカルな UI
2) RF-service
a) V,N の内部属性様式化の機能
b) M,C の様式化の機能
c) 言語依存活用辞書
3) CB-service
a) 言語依存意味辞書
4) RM-service
a) V,N の内部属性実体化の機能
b) M,C の実体化の機能
c) 言語依存活用辞書
5) GM-service
a) 線形化エンジン
b) 言語依存線形化辞書

3. 言語変換の実装－Word Processor

3.1 ワード・プロセッサー

言語変換の実装は、事実上一つのソフトウェア・プラットフォームの上で稼働するのが合理的である。それは、一種の word processor と考えてよい。つまり、前節で 0)で定義した word processor は 1)～5)までの機能のプラットフォームとしての役割を持つものである。若干の違和感はあるが、これを従来の通り「word processor」と呼ぶことにするが、その要件は、言語プロセッサーとでも呼ぶべきものである。

3.2 実装要件

1) 古代語・古典語の入力・表示・出力機能
ソフトウェア・キーボード
Unicode に依存しない文字コードとフォント
言語依存の入力・文字検索支援機能
2) キャレット位置の言語の自動認識
複数の言語が混在する環境となることが多く、言語依存の機能を実装するために、キャレット位置の文字列がどの言語化を識別する機能が必須である。
たとえば、ソフトウェアキーボートなども、カレット位置の言語によって自動的に切り替えられねばならない。
3) N,V parsing 機能
4) N-declension, V-conjugation 生成機能
一般に古代語、古典語では実例のない活用形の使用は良しとされない。しかしながら言語の双方向変換は、言語の理解のためには必須であるから、実装上、表に頼らない生成機能は必須である。
5) 辞書
辞書は、前節で論じたように、三種類のものが必要である。それは、線形化辞書(統語辞書), 活用辞書、意味辞書である。
古典語・古代語の場合、文脈自由な翻訳が可とされることが多いので、意味辞書は軽く、一意の検索ができるように作.ることが可能である。

4 実装の現状と今後の展望

4.1 現状

思いのほか手間取った、プラットフォームとしての Word Processor の実装がほぼ完了しつつある。この後、いよいよ言語変換の実装が本格化する。

4.2 対象言語

文脈自由、という点は現代語では大きな欠点となりうる。しかしながら、古典語・古代語においては、この点こそがむしろ利点となる。たとえば、I have a friend.を「私は友達を持つ」という表現こそが、古代の言葉の理解において必要な翻訳方なのであるから。

4.3 展望

今後も商用ペースに乗らなければ、あと10年ほどで完成することになろう。適用分野としては、発掘現場でのタブレット PC での利用や、言語のフィールド調査、開発途上国の国語教育の支援などが考えられる。
残念ながら、経済的に「役に立つ」システムではなさそうだが。