명제어로 표현된 한국어 문장의 개념분석

송 준용, 김 진형

한국 과학 기술원, 전산학과

요 약

본 논문은 명제어로 표현된 한국어 문장의 개념분석을 목표로 Expectation-based 방식의 제어구조를 개념분석에 이용하고 있다. 시스템의 입력은 명제어문장으로 격 Frame 과는 다른 구조인 Predicate form 이고, 시스템의 출력은 수정된 Conceptual Dependency(CD) form으로 R.C.Schank의 CD form 과는 약간 다른 형식을 갖고 있다. CD frame의 slot에 접속부사와 같은 표층구조의 구성요소가 표현되고 State 도 slot개념을 사용하여 표현하고 있다.

1. 서론

자연어처리분야는 기계번역과 기계이해분야로 크게 대별이 되고, 기계번역은 다시 Direct방식, Transfer방식, 그리고 Pivot 방식으로 나뉘어 진다. 이중에서 기계이해분야와 기계번역분야의 Pivot 방식에는 언어의 의미만을 표현하는 중간표현이 필요하다. 따라서 이러한 중간표현을 언어의 표층구조에서 얻어내기 위해서 개념분석을 하게된다.

본 논문에서는 자연어문장의 표층구조로부터 얻어진 명제어문장을 입력으로 해서 한국어문장의 개념분석을 하게 되는데, 명제어문장은 하나의 술어를 중심으로 구성된 리스트로 표현된다. 출력으로는 R.C.Schank의 Conceptual Dependency(CD)의 수정된 형태로 State 표현도 Action표현과 같이 frame 개념을 이용한다. 개념분석시스템은 한글lisp인 hlisp 을 이용하여 구현한다.

2. 한국어문장의 명제어표현

(1) 명제어표현을 도입한 배경

자연어상태의 한국어문장을 명제어로 바꾸기위해서는 자연어문장을 구성하는 여러 요소들과 그 요소들의 특징을 잃어버리지 말고 명제어문장에 표현해주어야 한다. 다만 명제어문장에서는 구성요소들의 순서와 특징을 표현하는 방식이 자연어문장과는 달라지게 된다. 일반적으로 한국어문장은 영어문장과는 달리 주어, 목적어, 동사의 순서로 문장이 구성된다. CD 이론에 의해서 개념분석을 하는 경우 동사의 개념구조가 그 문장의미를 대표하기때문에 문장에서 동사가 가장 늦게 나오는 한국어의 경우는 문장의 개념구조가 그 문장의 마지막까지와서야 결정이 된다. 이러한 이유로 문장을 단위로 해서 개념분석을 하는 경우, 문장의 왼쪽에서부터 오른쪽으로 한단어씩 개념분석과정을 거치기 때문에 동사이전에 나타난

각 단어의 개념구조를 동사를 만날 때까지 유지해야 한다는 단점이 있다. 또한, 동사의 개념구조내의 빈 slot 을 채울 단어의 개념구조들을 메모리내에서 찾아야 하는 overhead가 있다.

명제어를 사용하게된 또 다른 이유는 자연어 상태의 한국어문장을 직접 개념분석하기가 너무 복잡하다는 데 있다. 자연어문장을 그대로 입력으로 받는 경우, 형태소 분석과 구문 해석, 기타 의미적 해석과 대응어 처리등이 동시에 수행되어야 하는 어려운 점이 있다. 그러므로 한국어문장을 명제어와 같은 중간 언어로 바꾸어 줌으로써, 개념분석과정을 분리시키는 것이 좋다.
또한, 중간 언어인 명제어를 텍스트 Summarizing 시스템등의 자연어 처리 응용 분야에 이용할 수 있다는 장점이 있다.

(2) 명제어문장의 구조

명제어 표현은 문장의 동사나 형용사를 중심으로 해서 구성이 된다. 만일 한 문장에 동사나 형용사가 여러개 있을 경우에는 그 수만큼 명제어문장의 수가 결정이 된다. 따라서 명제어문장을 개념분석하게 되면, 개념분석 결과는 State나 Primitive ACT하나로 구성된 CD형태가 된다. 즉, CD구조의 slot 내에 다른 State나 ACT 가 들어가지 않는다. 만일 slot 내에 다른 State나 ACT가 표현되어야 하는 문장이라면, 그 State나 ACT는 다른 명제어문장으로 표현된다. 다만, 두 명제어간의 관계는 한 쪽을 포함하는 명제어문장내에 연결 부사등으로 표현을 한다. 대등한 연결 형태의 중문인 경우도 첫번째 나오는 동사로 구성된 명제어문장내에 연결 부사 항목을 넣어서 표현을 한다. 여기서 연결 부사는 순수한 접속 부사뿐만 아니라 어말어미도 포함하는 넓은 의미를 내포하고 있다. 예를 들면, "-할수록 "은 " 그럴수록 "으로 "-하다면 "은 " 그렇다면 "등으로 바꿔서 표현을 한다.
또한, 수식 부사도 명제어문장에 그대로 표현이 된다. 따라서, 이러한 명제어문장이 개념분석을 거쳐서 CD로 표현이 된다면, CD구조내에 연결 부사라든지 수식 부사가 slot을 차지하는 형태가 된다. 이러한 표현은 표층구조의 요소가 개념구조내에 남아 있기 때문에 순수한 의미만을 뜻한다고는 말할 수 없으나 명제어 표현이 텍스트를 표현하기 위해 확장이 된다면, 그에 따라 개념구조간의 연결관계도 생각해야 하므로 이러한 표현 방식을 사용한다. 기존의 CD이론에서는 이를 Causal Bond 로 표현하였으나, 이는 순수한 의미만을 표현하기 위해 도입한 것으로 실제적인 처리면에 있어서는 이를 이해하기 위한 overhead만을 추가할 뿐이다.

명제어 표현이 텍스트를 표현하기 위해서는 대응어 처리결과를 표현할 수 있어야 한다. 따라서 명사에 대한 대응어는 Reference 의 R 자로 표현을 해주고, 동사에 대한 대응어는 원래의 동사를 찾아서 표현을 해준다. 그리고 문장에 대한 대응어, 여기서는 명제어문장단위의 동사나 형용사 하나로 구성된 문장을 말하는데, 이러한 대응어는 명제어 문장번호를 사용하여 표현을 해 준다.

자연어문장이 명제어문장으로 바뀐 예를 들어 보면, 다음과 같다.

(자연어 문장)

" 철수는 집에 와서, 쥬스를 마셨다.
그 쥬스는 매우 달았다. 그런데 그것은 피로회복에 좋았다. "

(명제어 문장)

(N10	오다	(철수	R1)	(집 P2)	(과거 T1)	그리고나서)
(N20	마시다	R1		(쥬스 R3)	T1	)
(N30	달다	R3		(매우 AA)	T1	그런데)
(N40	좋다	N30		(피로회복 R4)	T1	)

<표-1> 명제어 문장의 표현 예

위에서 N10, N20, N30, N40은 명제어문장을 대표하는 문장번호이고, "오다", "마시다"는 동사, "달다", "좋다"는 형용사이다.
(철수 R1)이나 (쥬스 R3), (피로회복 R4)는 명사에 대한 표현으로 명사가 처음 나오는 경우, 나중에 대명사 표현을 위해서 R1, R3, R4가 첨가되어 리스트로 표현이 된다. 이 Rn의 표현은 동사에 대한 개념구조의 ACTOR와 OBJECT slot을 채우기 위한 정보로 이용되고, STATE에 대한 개념구조에서는 "주체"와 "대상" slot을 채우기 위한 정보로 이용된다.
"배우"와 같은 수식 부사는 CD에서는 Action Aider의 역할을 하기 때문에 AA로 표시를 해서 ACT나 State를 변형시키는 역할을 해준다.
(집 P1)은 "장소"의 의미를 갖는 명사로 PLACE의 P자로 표현을 한다. 이는 개념구조의 DIRECTION slot을 채우기 위한 정보로 이용된다. 이외에 (명사 An)에서 A는 Accepter의 약자로 개념구조의 RECIPIENT slot을 채우기 위한 정보로 이용되고, (명사 In)은 INSTRUMENT slot 채우는 데 사용된다.
이때 Rn, Pn, An, In의 표현중에서 Pn은 명사자체의 의미에 의한 표현이기 때문에 다른 명제어문장내에 명사에 대한 대응어로 그대로 사용될 수 있으나, Rn, An, In의 경우에는 문장에 따라 그 명사의 역할이 틀려지므로 서로 바꿔어 표현될 수 있다. 즉, 위와 같은 문장의 R1이 다른 명제어문장에서 Accepter로 쓰였을 경우, (R1 A1)으로 표현이 된다. 이와 같은 Rn, Pn, An, In의 표현은 자연어문장, 즉 표층구조의 조사에 따라서 결정된다.
(과거 T1)의 경우는 시스템에서 정한 표현으로 시제가 과거일 때는 항상 T1으로 표시가 된다. 시제와 관련된 표현으로 (과거부정 T2), (현재부정 T3) (미래 T4), (미래부정 T5), (무시제 T6)가 있고, 시제표현이 없는 경우는 "현재" 시제로 본다. 이때 (무시제 T6)은 역사적 현재나 진리, 일반적 사실 등의 표현을 위해 사용된다. 진행형의 문장이나 의문형의 문장일 경우에는 이러한 시제표현외에 별도로 "*진행*"과 "*의문*" 표시가 시제표현 바로 뒤에 나온다.
그리고 "그리고나서", "그런데" 등은 표층구조의 어말어미나 접속부사에 대한 표현으로 명제어문장간의 Causal Bond와 접속관계를 설명한다.
끝으로, 마지막 명제어문장내의 N30은 N30으로 대표되는 명제어문장대신에 쓰인 대응어이다.
이상에서 살펴본 명제어문장의 표현구조를 집약시켜 보면 다음과 같다. 이때 명제어문장을 구성하는 각 요소들은 한국어의 구조가 일항, 이항, 삼항진술로 구성된다는 데 근거를 둔다.

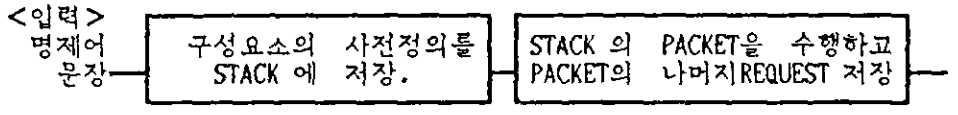
```
(명제어문장번호   동사 | 형용사  { ACTOR | 주체 }  { [OBJECT | 대상] }
                  { [PLACE ] }   { [ACCEPTER] }   { [INSTRUMENT] }
                  { [AA] }         [시제표현]       [접속부사]        )
```

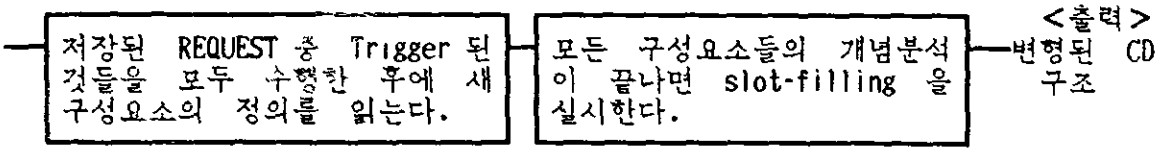
<참고> [] : 생략가능, { } : 반복, | : 선택

<표-2> 명제어 문장의 FORMAT

이 중에서 [PLACE], [ACCEPTER], [INSTRUMENT], [AA]의 순서는 바뀔 수 있으나, 나머지 항목들은 정해진 순서로 표현되며, [ACCEPTER], [INSTRUMENT]는 형용사를 갖고 있는 명제어문장내에는 표현되지 않는다.

3. 명제어문장의 개념분석





<그림-1> 개념분석 DIAGRAM

(1) 개념분석 시스템의 개요

이 시스템의 전체적인 흐름은 그림-1에 나타나 있다. 명제어문장의 구성요소인 단어나 리스트를 읽어서 사전에서 Entry를 찾아 PACKET 즉, 단어정의를 STACK에 넣는다. PACKET의 첫 Request가 Trigger 되면, 그 Request의 Action을 수행한다. 더 이상 Trigger 된 Request가 없을 때는 그 PACKET을 구성하는 나머지 Request 들을 TRIGGERED 리스트에 저장한다. 새로운 입력 PACKET에 의해 TRIGGERED 리스트내의 Request가 Trigger 되면, 그 Request의 Action을 수행하여 리스트내의 활성화된 모든 Request 들을 제거한다. 이러한 방법으로 명제어문장의 모든 구성요소에 대한 개념분석이 끝나면, 만들어진 개념구조내의 빈 slot을 채워서 출력한다.

이 시스템에서는 명제어 텍스트를 한 단위로 하여 전체 시스템을 총괄하는 최상위 루틴에서 각 명제어문장을 개념분석하는 파싱 루틴을 수행시켜 개념분석을 한다.

파싱 루틴은 명제어문장을 구성하는 각 요소들의 개념구조를 출력시키고 각 문장을 대표하는 동사나 형용사의 개념구조의 빈 slot에 이러한 요소들의 개념구조들을 채워 넣는다.

이 파싱 루틴의 기능을 간단히 살펴 보면 다음과 같다.

1) 단어가 사전에 있는지의 여부를 검사하여 단어의 PACKET을 STACK에 넣는다. 사전에 없으면, STACK 내의 다음 PACKET을 수행시킨다.

2) STACK 내에서 Trigger 된 PACKET의 Request 들을 수행시키고, PACKET 내의 나머지 Request 들을 TRIGGERED 리스트에 저장한다. Trigger 된 것이 없을 때에는 TRIGGERED 리스트에 저장된 Request 들을 STACK에 더한다. 여기서 Request가 Trigger 되었다는 뜻은 Request의 TEST부분이 true가 되었음을 의미한다.

3) ASSIGN Clause를 수행시키고, 각 변수와 값들을 출력한다.

4) 명제어문장을 대표하는 개념구조내의 빈 slot에 적당한 개념구조들을 채워 넣는다. 이를 위해서 패턴 변수를 검사하는 함수와 CD Frame, 즉 (HEADER (ROLE (*VAR* id)) ...)의 각 구성 요소들을 검색할 수 있는 함수들이 있다.

파싱 루틴이외에 부수적으로 사용된 기능들로는 사전에 단어를 정의하기 위한 함수와 출력형태를 보기좋게 지정하기 위한 CONCEPT-PRINT 함수가 있다.

(2) 사전의 구성

사전의 각 단어가 갖고 있는 정보에는 품사 정보와 개념구조에 관한 정보가 있고, 동사나 형용사와 같은 술어에는 빈 slot을 채우기 위한 Request에 관한 정보가 추가되어 있다. 그 외에 각 단어의 성격에 따라 필요한 정보들이

```
(단어정의    ***
    ((ASSIGN     Variable    Value
                 Variable    Value
                   ..............)
    (NEXT-PACKET
         ((TEST  ( ......))
          (ASSIGN   Variable   Value
                      ............)
          (NEXT-PACKET
              ......................]
```

<그림－2> 사전의 단어 정의 FORMAT

그러면 이러한 FORMAT에 의해 실제로 <표－1>에 나타난 명제어문장의 구성요소중 "철수"와 "오다"의 단어정의에 관해 알아 보자.

```
(단어정의    철수
    ((ASSIGN    개념구조   ' (사람   (이름   (철수)) (성별   (남성)) )]
                품사       ' 명사 ]

(단어정의    오다
    ((ASSIGN    품사       ' 동사
                개념구조   '  (PTRANS  (ACTOR  ?P1)
                                (OBJECT ?P1)
                                (TO     ?P2)
                                (FROM   ?P3)
                                ( 시제  ?P4)
                                ( 수식부사  ?P5)
                                ( 연결부사  ?P6))
                P1-P6      NIL)
    (NEXT-PACKET
         ((TEST  (equal  품사  ' 명사))
          (ASSIGN   P1   개념구조)
          (NEXT-PACKET
               ((TEST  (equal  품사  ' 명사))
                (ASSIGN  P2  개념구조)
                (NEXT-PACKET
                     ((TEST  (equal  시제  ' 과거))
                      (ASSIGN  P4  시제)
                      (NEXT-PACKET
                           ((ASSIGN  P6  ' 그리고나서]
```

(3) 개념분석 결과

<표－1>의 명제어문장을 입력으로 했을 때의 개념분석 결과에 대해서 알아 보자.

```
입력 :  (N10  오다  ( 철수  R1)  (집 P2)  ( 과거  T1)   그리고나서)
출력 :  (PTRANS
          (ACTOR (사람   (이름   (철수)) (성별   (남성))  ))
          (OBJECT ( 사람   (이름   (철수)) (성별   (남성)) ))
          (TO   ( 사물   (형태   (집)) ))
```

```
                ( 시제    과거)
                ( 연결부사    그리고나서))

입력 :  (N20  마시다  R1  ( 쥬스  R3)   T1 )
출력 :  (INGEST
              (ACTOR  ( 사람   (이름   (철수))  ( 성별   (남성)) ))
              (OBJECT ( 사물   (형태   (물))  ( 종류   (쥬스)) ))
              (TO   입)
              (FROM 손)
              ( 시제  과거))

입력 :  (N30  달다  R3   (매우   AA)   T1  그런데  )
출력 :  (STATE
              (TASTE  +8)
              ( 주체   (사물   (형태   (물))  ( 종류   (쥬스)) ))
              ( 시제  과거)
              ( 수식부사  매우)
              ( 연결부사  그런데))

입력 :  (N40  좋다  N30   ( 피로회복  R4)   T1)
출력 :  (STATE
              (MENTAL-STATE   +7)
              (PHYSICAL-STATE   +8)
              ( 주체  (STATE
                           (TASTE   +8)
                           ( 주체   (사물   (형태   (물))(종류   (쥬스)) ))
                           ( 시제  과거)
                           ( 수식부사  매우)
                           ( 연결부사  그런데)) )
              ( 대상  (PHYSICAL&MENTAL-STATE:GROWTH    (내용   (피로회복)) ))
              ( 시제  과거))
```

4. 결론

자연언어 처리분야에서 언어의 기계적 이해 분야의 중요성이 강조되고 있는 오늘날, 우리에게 닥친 가장 커다란 과제는 한국어 PARSER를 구현하는 일이다. 이러한 현실에서 본 논문은 PARSER의 개념분석과정이 중간 언어를 통하여 이루어 져야 한다는 것을 주장하고 있다.

현재, 한국어의 형태소 분석과 구문 해석에 관한 연구가 활발히 진행되고 있고, 대응어 처리 문제도 부분적으로 연구가 되고 있기 때문에, 이러한 중간 언어를 만들 수 있는 능력을 조만간 갖추게 될 것이다.

본 논문에서는 여러가지 형식의 중간 언어가 개발될 수 있다는 전제하에 술어 하나로 구성된 중간 표현을 가정하였다. 다만, 명사구에 대한 명제어 표현 방법과 이를 개념분석 하는 과정에 대해서는 여기서 언급을 하고 있지 않지만, 이 문제는 중간 언어 표현의 확장을 통해서 얼마든 지 극복할 수 있을 것이다.

<참고 문헌>

1) [CHOI83] 최 창렬. 한국어의 의미 구조. 한신 문화사, 1983.

2) [NLLT82] 남 기심, 이 정민, 이 홍배. 언어학 개론. 탑 출판사, 1982.

3) [DAGC84] Fum,D., Guida,G. and Tasso,C. A Propositional Language For Text Representation. In Bara,B.G. and Guida,G.(Eds.), Computational Models of Natural Language Processing. Amsterdam:North Holland,1984.

4) [DYER83] Dyer,M.G. In-Depth Understanding : A Computer Model of Integrated Processing for Narrative Comprehension. The MIT Press,1983.

5) [HARR85] Harris,M.D. Introduction to Natural Language Processing. Reston,Virginia : Reston,1985.

6) [RIES75] Riesbeck,C.K. Conceptual Analysis. In Schank,R.C.(Ed.), Conceptual Information Processing. Amsterdam : North Holland,1975.

7) [SAML76] Samlowski,W. Case Grammar. In Charniak,E. and Wilks,Y.(Eds.), Computational semantics. Amsterdam : North Holland, 1976.

8) [SCAB77] Schank,R.C. and Abelson,R.P. Scripts,Plans,Goals and Understanding Hillsdale,N.J. : Lawrence Erlbaum Associate, 1977.

9) [SCHA73] Schank,R.C. Identification of Conceptualizations Underlying Natural Language. In Schank,R.C. and Colby,K.M.(Eds.), Computer Models of Thought and Language> San Francisco : W.H.Freeman and Company,1973.

10) [SCRI81] Schank,R.C. and Riesbeck,C.K. Inside Computer Understanding : Five Programs plus Miniatures. Hillsdale,N.J. : Lawrence Erlbaum Associate,1981.