Kant 시스템에서의 한국어 생성을 위한 언어 정보의 구축

윤 덕 호†

요 약

KANT(Knowledge-based Accurate Natural language Translation) 시스템 생성 엔진을 위한 한국어 언어 정보를 구축하였다. KANT 시스템은 언어 중립적인 생성 엔진을 갖고 있기 때문에 한국어 언어 정보의 구축은 사실상 한국어 생성 모듈의 개발을 의미한다. 구축된 언어 정보는 개념별 한국어 대응 규칙, 범주별 한국어 대응 규칙, 한국어 사전 및 템플리트 선언, 한국어 문법 규칙, 한국어 어휘 유형, 한국어 어휘 규칙, 한국어 다시 쓰기 규칙 등으로 구성된다. 구축된 언어 정보를 이용해 KANT 시스템 개발 측이 준비한 118 문장 분량의 중간 언어 표현로부터 106 문장을 올바르며 완전한 한국어 문장으로서 생성하였다.

Construction of Korean Linguistic Information for the Korean Generation on KANT

Deok-Ho Yoon†

ABSTRACT

Korean linguistic information for the generation module of KANT(Knowledge-based Accurate Natural language Translation) system was constructed. As KANT has a language-independent generation engine, the construction of Korean linguistic information means the development of the Korean generation module. Constructed information includes concept-based mapping rules, category-based mapping rules, syntactic lexicon, template rules, grammar rules based on the unification grammar, lexical rules and rewriting rules for Korean. With these information 106 sentences were successfully and completely generated from the interlingua functional structures among the 118 test set prepared by the developers of KANT system.

1. 서 론

KANT는 미국 CMU(Carnegie Mellon University) 대학에서 80년대 후반 이후 지속적으로 연구 개발 및 확장 작업을 수행하고 있는 중간 언어 방식의 다국어간 기계 번역 시스템[1, 2]이다. KANT 시스템은 중간 언어 표현을 매개로 하여 분석기와 생성기를 완전히 분리하는 방식으로 다국어간 기계 번역에 접근하고 있으며[3], 언어 중적인 분석 엔진과 생성 엔진을 개발하고 개별 언어에 대한 언어 정보를 이 엔진에 부가함으로써 언어 중립적 엔진이 바로 해당 언어의 분석기 혹은 생성기로서 동작하게 하는 접근 방법을 취하고 있다[4].

본 연구에서는 한국어 생성을 위하여 필요한 제반 언어 정보를 아래와 같이 구축함으로써 KANT 시스템에서의 한국어 생성을 가능하게 하였다.

- 개념별 한국어 대응 규칙 : 12 개념 계열, 420 규칙
- 범주별 한국어 대응 규칙 : 7 범주, 61 규칙
- 한국어 사전 : 15 템플리트 정의를 이용한 424 항목
- 한국어 문법 규칙 : 19 문법 기호와 73 규칙
- 한국어 어휘 유형 : 115 어휘 유형

† 정 회 원 : 한남대학교 정보통신공학과 교수
논문접수 : 1999년 7월 14일, 심사완료 : 1999년 11월 12일

- 한국어 어휘 규칙 : 166 어휘 규칙
- 다시 쓰기 규칙 : 68 항목

위와 같은 언어 정보의 구축은 KANT 시스템을 개발 관리하고 있는 CMU 대학의 LTI 연구소에서 제공한 118 문장 규모의 테스트용 중간 언어 표현에 대한 적용 실험과 병행하여 이루어졌으며, 그 결과 테스트 집합의 118 문장 가운데 106 문장에 대해 올바르며 완전하게 생성 결과를 낼 수 있었다.

2. 중간 언어 방식의 다국어간 기계 번역 시스템 KANT

KANT는 1980년대 후반 미국 CMU 대학의 Carbonell, Nyberg 등에 의하여 발표된 이래, 꾸준한 연구 개발과 확장이 이루어지고 있는 중간 언어 방식의 다국어간 기계 번역 시스템이다. 중간 언어 방식(Interlingual Approach)은 (그림 1)과 같이 언어 중립적인 중간 언어(Interlingua) 표현을 매개로 원시 언어 분석과 목표 언어 생성의 과정을 분리한다. 이에 따라 언어당 하나씩의 분석기와 생성기만으로 언어간 번역이 가능하여 언어쌍별로 별도의 변환기를 필요로 하는 변환 방식(Transfer Approach)에 비하여 세 언어 이상을 상호 번역하는 다국어간 번역 시스템 개발을 편리하게 하며 특히 언어적 성격이 크게 달라 변환의 부담이 큰 언어쌍을 다수 포함하는 경우에 적합하다.

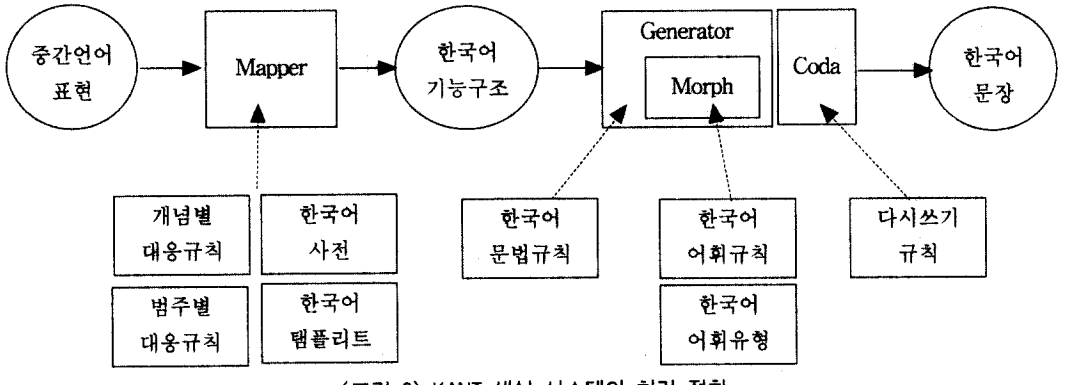
(그림 1) 중간 언어 방식의 기계 번역

또한 KANT 시스템에서는 분석기와 생성기 역시 개별 언어별로 따로 구현하는 대신 언어 중립적인 범용 엔진 형태로 구현하면서 이 엔진에 개별 언어에 대한 언어 의존적 정보를 제공하고, 이 언어 정보에 따라 분석이나 생성 대상 언어가 달라지도록 하는 접근 방법을 취했다. 이러한 접근 방법은 시스템 차원의 언어 의존적 요소를 줄임으로써 시스템의 이식성을 높이는 한편 새로운 언어의 추가와 관련한 확장 작업의 부담을 최소화하여 본격적인 다국어간 번역 시스템의 개발을 할 수 있게 한다.

생성 대상 언어를 한국어라 할 때 KANT의 생성 시스템의 처리 절차 및 정보 흐름은 (그림 2)와 같으며 분석 시스템은 이와 반대의 처리 절차 및 정보 흐름을 가진다[5].

먼저 Mapper 모듈은 생성기의 입력 정보인 문장의 중간 언어(Interlingua) 표현을 한국어 기능 구조(F-Structure)[6]로 변환한다. 문장 성분의 의미 역할을 중심으로 문장을 표현하는 중간 언어 표현과 달리 기능 구조는 문장 성분의 문법 기능을 중심으로 문장을 표현한다. 따라서 이 변환 작업에는 의미 역할과 문법 기능 사이의 대응 관계를 알려주기 위한 한국어 대응 규칙(Mapping Rules) 정보가 필요하다. Mapper 모듈은 또한 중간 언어인 개념 표현으로 나타난 문장 성분을 한국어 어휘 성분으로 대치하는 작업을 함께 수행하며, 이 때 한국어 사전(Lexicon) 정보가 이용된다.

Generator 모듈은 Mapper에 의하여 생성된 한국어 기능 구조에서 한국어 문장을 생성한다. 즉 Generator는 기능 구조에 나타난 문법 기능들에 단일화 문법에 기반한 한국어 문법 규칙을 적용하여 문장의 통사 구조를 결정하고, 결정된 통사 구조의 각 단말 항목에 대



(그림 2) KANT 생성 시스템의 처리 절차

해 Morph 모듈을 호출하여 한국어 문자열이 생성되도록 한다. 이 때 기능 구조 단말 항목에 대해 한국어 사전의 어휘 항목 정보와 함께 Mapper나 Generator가 부여한 각종 속성 정보가 전달되며 Morph 모듈은 한국어 어휘 규칙을 이용하여 이 정보들을 처리함으로써 목표 언어의 각종 언어 현상을 반영한 출력 문자열을 생성한다.

Generator 모듈에서 생성된 목표 언어 문자열은 문자열 패턴 매치 수준의 고쳐 쓰기 처리를 수행하는 Coda 모듈을 거쳐 최종 출력으로 확정된다. Coda는 문장 부호, 음운 현상 등의 처리에 이용하는 외에 아직 Generator 모듈의 언어 정보로 일반화되지 못한 예외적 현상의 처리에도 이용한다.

현재 KANT 시스템이 분석 가능한 언어는 영어 뿐이며, 생성 가능한 언어는 불어, 독일어, 이탈리아어, 스페인어, 포르투칼어, 러시아어, 일본어, 터키어 등이고 이번 연구로 한국어도 생성이 가능하게 되었다. 아직까지는 KANT 시스템이 영어를 더 많은 언어로 번역하는 데 주력하고 있지만 곧 다른 언어에 대한 분석기를 개발하여 다대다 번역이 가능한 시스템으로 확장에 나설 것으로 보인다. 이 경우 한 언어에 대한 분석기가 새로 개발될 때마다 그 언어로부터 생성기가 이미 개발된 모든 언어로의 번역이 가능하게 된다.

3. Mapper 모듈을 위한 한국어 언어 정보의 구축

생성기의 첫 단계인 Mapper 모듈[7]은 중간 언어(Interlingua)로 표현된 문장 정보를 목표 언어의 기능 구조(Target F-Structure)로 변환하는 역할을 한다. 본 연구에서는 한국어 대응 규칙 정보와 한국어 사전 정보를 구축하여 Mapper 모듈에 제공함으로써 중간 언어 표현을 한국어 기능 구조로 변환할 수 있게 하였다.

이러한 변환 과정과 변환에 사용된 정보를 예를 들어 간단히 살펴보도록 하자.

"The front hydraulic line provides fluid pressure to the hydraulic unit."

위의 영어 문장을 KANT 분석기가 분석하여 얻은 중간 언어 표현은 아래와 같다. 참고로 KANT의 분석 엔진 및 생성 엔진은 현재 모두 LISP 언어로 구현되어 있으며, 따라서 본 연구에서 사용한 모든 자료는 LISP 언어에 적합한 리스트 형태로 표현된다.

```
(*A-PROVIDE
  (PUNCTUATION PERIOD)
  (TENSE PRESENT)
  (MOOD DECLARATIVE)
  (ARGUMENT-CLASS AGENT+THEME)
  (AGENT
        (*O-HYDRAULIC-LINE
              (REFERENCE DEFINITE)
              (NUMBER (:OR SINGULAR MASS))
              (ATTRIBUTE (*P-FRONT))))
  (Q-MODIFIER
        (*Q-RECIPIENT_TO
              (ROLE RECIPIENT)
              (CASE (*K-TO))
              (OBJECT
                    (*O-HYDRAULIC-UNIT
                          (NUMBER SINGULAR)
                          (REFERENCE DEFINITE)))))
  (THEME
        (*O-FLUID-PRESSURE
              (REFERENCE NO-REFERENCE)
              (NUMBER (:OR SINGULAR MASS))))
```

위의 표현에서 *O-HYDRAULIC-LINE, *A-PROVIDE, *P-FRONT 등은 어휘 항목의 중간 언어 표현으로 이용되는 개념 표현(conceptual representation)이며 객체를 나타내는 O-계열, 동작을 나타내는 A-계열, 보조 속성을 나타내는 P-계열 등 12 계열로 나뉜다. 중간 언어 표현은 (중심개념 (속성 값) ... (속성 값)), 즉 하나의 중심 개념에 0개 이상의 (속성 값) 집합이 부가된 리스트 형태로 구성된다. 속성에는 AGENT, THEME, Q-MODIFIER 등의 의미역 속성, ARGUMENT-CLASS 등의 의미역 보조 속성, 그리고 NUMBER, TENSE, PUNCTUATION 등의 모달(modal) 속성 등이 있으며, 의미역 속성의 값은 하위 문장 성분에 대한 내포된 중간 언어 표현이고, 의미역 보조 속성이나 모달 속성의 값은 속성별 값 집합에 속하는 원소이다.

위의 표현은 Mapper의 처리를 거쳐 아래와 같은 한국어 기능 구조로 변환된다.

```
((OBJ  ((ENDING CONSNANT)
        (CAT N)
        (ROOT "유압")
        (POST-EXIST +)))
 (ROOT "제공-하")
 (CAT V)
 (VERBAL +)
```

```
(XADV-QMOD ((POST RO)
            (POST-EXIST +)
            (ENDING VOWEL)
            (CAT N)
            (ROOT "유압 장치")))
(MOOD DECLARATIVE)
(FORM FIN)
(TENSE PRESENT)
(INF-TYPE OBJECT)
(SUBJ ((POST-EXIST +)
       (ROOT "유압선")
       (CAT N)
       (ENDING CONSNANT)
       (NV-DET-ATT ((ENDING CONSNANT)
                    (CAT N)
                    (ROOT "앞쪽")
                    (POST-EXIST +)))))
(PUNCTUATION PERIOD))
```

기능 구조는 ((속성 값) ... (속성 값)), 즉 1개 이상의 (속성 값) 쌍의 집합을 나타내는 리스트 형태를 가진다. 기능 구조는 중간 언어 표현과 달리 문장 통사 구조의 정보 표현 형태로서 언어 의존적이다. 위의 예에서 보듯이 문장 성분의 의미 역할은 SUBJ, OBJ 등의 문법 기능으로 대치되었으며, 개념 표현 역시 "유압선", "제공-하", "앞쪽" 등과 같은 한국어 특유의 어휘 항목으로 대치되었음을 알 수 있다. 부수적 속성 역시 CAT, ENDING, VERBAL 등 통사적, 어휘적인 정보가 상당수 나타나고 있다.

Mapper는 중간 언어 표현을 특정 언어의 기능 구조로 변환하기 위하여 언어별로 구축된 대응 규칙 정보를 이용한다. Mapper가 이용하는 대응 규칙은 (node 규칙이름 키워드$_1$ 값$_1$ ... 키워드$_n$ 값$_n$)의 형태를 가지며 키워드$_i$에 따라 값$_i$는 조건이 되기도 하고 처리절차가 되기도 한다. 키워드의 종류와 기능을 간단히 소개하면 아래의 표와 같다.

예문의 경우 각종 조건 검사 결과 가장 먼저 적용되는 규칙은 :encodes 항이 예문 중간 언어 표현의 중심 개념에 부합되는 아래의 규칙이다. 이 규칙에서 먼저 :rules 항에 의해 새로운 기능 구조가 생성되고 :parent 항에 의해 verb라는 이름의 규칙에 의한 추가적인 처리가 행해지게 된다.

```
(node "A-PROVIDE"
      :parents verb
      :encodes (*A-PROVIDE)
      :rules ((:lex "jegongha-verb")))
```

:rules 항에서는 :lex 항에 의해 생성되는 기능 구조를 예문 전체에 대한 기능 구조로 삼을 것을 지시한다. :lex 항에 의해 사전이 검색되며 아래와 같은 사전 정보가 찾아진다.

```
(cha-verb "jegongha-verb" ((root "제공-하")))
```

이 사전 정보는 어휘 항목에 대한 기능 구조가 'cha-verb' 템플릿을 적용하여 얻어지는 기능 구조와 '((root "제공-하"))'라는 기능 구조의 단일화로 얻어질 수 있음을 알려 주고 있다. 이처럼 모든 어휘 항목 사전 정보는 하나의 템플릿에 속하는데 'cha-verb' 유형의 템플릿 정보는 아래와 같다. 따라서 "jegongha-verb"의 기능 구조는 '((root "제공-하"))'와 '((cat v) (verbal +))'의 단일화 결과인 '((root "제공-하") (cat v) (verbal +))'가 된다.

```
(soft-template cha-verb ((cat v) (verbal +)))
```

:rules 항에 의한 기능 구조 생성이 끝나면 :parent 항에 지정된 verb 규칙을 생성된 기능 구조에 적용한다. verb

키워드	기 능
:encodes	중간언어표현의 중심개념이 지정된 값집합에 속하는지를 검사한다
:parent	중간언어표현에 대해 지정된 이름의 규칙도 함께 적용한다.
:rules	지정된 방법으로 중간언어표현에 대응되는 새 기능 구조를 생성한다.
:test	지정된 조건을 검사하여 만족되면 지정된 처리를 수행한다.
:sem	지정된 방법으로 중간언어표현 내부를 검사한다.
:syn	지정된 방법으로 기능구조 내부를 검사한다.
:lex	지정된 어휘 항목을 사전에서 찾아 대응 기능구조를 생성한다.
:ir	중간언어표현에 지정된 속성이 있는지 검사하고 속성값을 반환한다.
:and, :or, :not	간단한 조건들을 엮어 복잡한 조건을 만든다.
:map	지정된 중간언어표현 속성값들을 재귀적 방법으로 기능구조로 변환해 현재 기능 구조에 지정된 속성값으로 부여한다.
:add	새로운 기능구조를 만들어 현재 기능 구조와 단일화시킨다.

규칙은 아래와 같이 정의되어 별도의 파일에 저장된 일련의 항들을 적용할 수 있게 되어 있으며 다시 일반 범주를 위한 general 규칙의 적용을 지시한다.

```
(node verb
      :parents general
      :rules("/afs/cs/project/cmt-48/kant-korean
            /ver1/map/kverb.map"))
```

별도의 파일에 저장된 규칙 항들은 조건과 처리로 구성된 :test 항의 나열로서 작성되어 있으며, 예문과 관련하여 적용되는 가장 중요한 항은 기능 구조에서의 문장 성분을 밝혀 주는 아래의 항이다.

```
(:test (:sem (:and (ARGUMENT-CLASS AGENT+THEME)
                   (:not (TOPIC-ROLE THEME))))
    :map       ((AGENT SUBJ) (THEME OBJ)))
```

위의 항은 중간 언어 표현의 ARGUMENT-CLASS 속성값과 TOPIC-ROLE 속성값을 검사하여 조건이 만족되면 중간 언어 표현의 AGENT 속성값과 THEME 속성값을 기능 구조로 변환하여 현재의 기능 구조에 SUBJ 속성값과 OBJ 속성값으로 대응시킨다. 즉, Mapper가 중간 언어 표현 안에 기술된 AGENT 및 THEME 속성값을 변환하여 얻은 기능 구조를 각각 F_{AGENT}, F_{THEME}이라 한다면 문장 전체에 대한 기능 구조는 '((root "제공-하") (cat v) (verbal +) (SUBJ F_{AGENT}) (OBJ F_{THEME}))'으로 확장된다. F_{AGENT}, F_{THEME}는 해당 중간 언어 표현의 중심 개념, *O-HYDRAULIC-LINE와 *O-FLUID-PRESSURE를 출발점으로 삼아 마찬가지의 재귀적 처리 과정을 거쳐 얻어지게 된다.

주요 문장 성분 외의 기능 구조 요소들도 다양한 과정을 거쳐 얻어지는데 예문의 경우에서 가능 구조의 부수적 속성 정보가 생성되는 과정을 보여주는 간단한 예를 하나 소개하면 아래와 같다.

```
(:test  (:syn  (MOOD :undefined)
        :sem (MOOD :defined))
  :add ((mood (:ir MOOD))))
```

위의 항은 MOOD 속성값이 중간 언어 표현에 존재하고 생성 중인 기능 구조에는 다른 이유로 아직 정의된 적이 없는가를 검사한다. 예문의 경우 검사를 만족시키므로 먼저 중간 언어 표현의 MOOD 속성값 DECLARATIVE를 얻어내 '((mood DECLARATIVE))'라는 기능 구조를 만들고 이 구조를 문장 전체의 기능 구조와 단일화 시킨다. 많은 부수적 속성 정보가 유사한 규칙에 의하여 얻어지게 된다.

이와 같이 Mapper는 대응 규칙의 검색 및 적용 과정을 통하여 문장 전체에 대한 중간 언어 표현을 대응되는 생성 대상 언어의 기능 구조로 변환한다.

본 연구에서는 한국어 기능 구조 생성을 위한 정보로서 예시한 규칙들을 포함하여 427개의 규칙을 작성하였으며, 이 가운데 420개는 :encode 항을 갖는 개념 차원의 규칙이며 7개는 :parent로 지시되는 범주별 규칙이다. 7개의 범주별 규칙은 그 내부에 모두 61개의 조건-처리 쌍으로 구성된 :test 항을 갖는다. 한편 :lex 항에서 참조하는 사전 정보를 424개의 어휘 항목에 대하여 구축하였으며 15 유형의 템플릿 정보도 함께 구축하였다.

4. Generator 모듈을 위한 한국어 언어 정보의 구축

Generator[8]는 Mapper가 생성한 기능 구조에서 목표 언어 문장을 생성한다.

이를 위하여 우선 Generator는 단일화 문법에 기반한 문법 규칙을 기능 구조에 적용함으로써 문장의 통사 구조를 결정한다. 문법 규칙은 문맥 자유 문법(CFG) 형태의 기본 규칙에 기능 구조 통합 조건이 부가된 형태로 작성된다. 이들 문법 규칙은 생성용 규칙이므로 좌변에서 우변으로 전개되며, 기능 구조 통합 조건은 통합이 아닌 분리의 과정을 알려주는 역할을 한다. 아래 몇 개의 문법 규칙을 예로서 소개하기로 한다.

```
(<start> ==> (<korean-sentence-phrase>)
          ((x1 = x0)))
(<korean-sentence-phrase> ==> (<main-vp> <punctuation>)'
          ((x2 < (x0 PUNCTUATION))
           (x1 = x0)))
(<punctuation> --> (period)
          ((x0 =c period)))
(<main-vp> ==> (<vp>)
          (((x0 TOPIC) =c SUBJ)
           (x1 = x0)
           ((x1 SUBJ IS-TOPIC) = +)))
(<vp> ==> (<np-conj> <vp>)
          ((x1 < (x0 SUBJ))
           ((x1 CASE) = SUBJ)
            (x2 = x0)))
```

위에 예시한 문법 규칙에 나타난 기능 구조 통합 조건

의 의미를 간단히 소개하겠다.

(x1 = x0) : 우변 첫 항의 기능 구조가 좌변의 기능 구조와 같다는 의미, 엄밀하게는 통합 가능하다는 의미이다. 생성 단계이므로 좌변 기능 구조가 제시된 상태이며, 따라서 우변 첫 항의 기능 구조를 결정하는 역할을 한다. 물론 우변 첫 항의 기능 구조는 다른 기능 구조 통합 조건의 영향도 받으므로 좌변 기능 구조와 다소 달라질 수 있다.

((x1 CASE) = SUBJ) : 우변 첫 항의 기능 구조의 CASE 속성의 값으로 SUBJ를 부여한다. CASE 속성이 없었다면 신설되며 이미 있으면서 값이 SUBJ가 아니면 문법 규칙 적용에 실패하게 된다. 즉 이 항은 속성 정보 부여와 조건 검사의 역할을 함께 한다.

(x1 < (x0 SUBJ)) : 좌변 기능 구조의 SUBJ 속성값을 우변 첫 항의 기능 구조로 부여하게 된다. '<'이 '='과 다른 점은 좌변 기능 구조에서 사용하고 난 SUBJ 속성값을 제거하여 유사한 다른 규칙의 적용을 불가능하게 만든다는 점이다. 예를 들어, 위의 항목을 갖는 두 개의 문법 규칙이 있을 때 첫 규칙이 다른 검사를 통과하여 이 항목을 처리하게 되면 좌변 기능 구조의 SUBJ 속성값이 사라져 이후 다음 규칙이 적용되어 다른 조건이 다 만족되더라도 위의 항목에서 실패가 일어나게 된다. 이로 인해 규칙 사이의 순서가 결과에 중요한 영향을 미치게 되는 부작용을 낳게 되지만 KANT 생성 시스템은 효율적인 처리를 위해 이 방법을 채택하고 있다[8].

((x0 TOPIC) =c SUBJ) : 좌변 기능 구조의 TOPIC 속성값이 SUBJ라는 원소값인지 검사한다. 값이 정의되어 있지 않거나 다른 값이면 실패로 처리하여 규칙을 적용하지 않는다.

판명된 통사 구조의 단말 노드에 대해서는 아래와 같은 규칙을 두어 형태소 차원의 생성 작업을 수행하도록 하였다. (%)는 출력 문자열을 value 속성으로 가지면서 더 이상의 문법 적용을 않는 단말항을 나타내고 ko-gmorph는 한국어 형태소 생성을 위한 Morph 모듈의 호출을 나타내며 (ko-gmorph (x0))는 한국어 형태소 생성에 단말항의 기능 구조 정보가 전달되어 이용됨을 나타낸다.

```
(<verb> ==> (%)
        (((x0 CAT) =c V)
         ((x1 value) <= (ko-gmorph (x0)))))
```

각 단말 항목에 대하여 Generator 모듈에 의하여 호출되는 Morph 모듈은 단말 항목의 기능 구조 정보를 토대로 형태소를 생성한다.

Morph 모듈은 한국어 어휘 유형 정보에 따라 주어진 기능 구조를 분류해 나감으로써 하나의 어휘 유형을 결정하며, 이 어휘 유형에 대한 어휘 규칙의 적용을 통하여 기능 구조에 대응되는 형태소 출력 문자열을 결정한다.

3절 예와 비슷한 기능 구조가 주절 서술어 단말 항목에 아래와 같이 전달되었다고 가정하자. 적절한 예시를 위하여 negation 속성 및 possibility 속성이 추가된 형태로 가정한다.

```
((cat v) (root "제공-하") (negation +) (possibility +) .....)
```

Morph 모듈은 기능 구조의 부수적 속성값들을 토대로 어휘 항목의 어휘 유형을 결정하며, 이 때 이용되는 것이 아래와 같은 어휘 유형 정보이다.

```
(morph-form v          *          (cat v))
(morph-form v+active   v          (*not* (passive +)))
(morph-form v+act+neg  v+active   (negation +))
(morph-form v+an+poss  v+act+neg  (possibility +))
```

우선 주어진 기능 구조는 (cat v) 항목을 가지므로 v 유형이 되며 v 유형이면서 (passive +) 항목을 갖지 않으므로 v+active 유형이 된다. 이런 식으로 따라가면서 최종적으로는 동사의 능동-부정-가능형을 나타내는 v+an+poss 유형이 된다. 참고로 유형의 이름은 '-', '+' 등의 기호를 포함해도 무방하며 모두 연구 과정에서 지어준 이름이다. 어휘 유형 정보는 어떤 유효한 기능 구조에 대해서도 단일한 경로로의 추적이 가능한 트리 형태로 구성되어 있다.

각 단말 어휘 유형에 대해서는 대응되는 어휘 규칙을 두어 생성할 문자열을 지정하게 되어 있다. v+an+poss 유형에 대한 어휘 규칙은 아래와 같다. (*)는 기능 구조의 root 속성값 문자열을 나타내며 +s는 후위 접속 문자열 연산을 의미한다. 이 결과 얻어지는 출력 문자열은 "제공-하-을 수 없"이다.

```
(morph-rule  v+an+poss      ((*) (+s "-을 수 없")))
```

참고로 위의 문자열 외에도 어미에 대한 유사한 처리를 통해 아래의 어휘 규칙이 적용될 수 있다. 그런데 어미는 문법 규칙에서 root 속성값으로 "@nospace@"가 부여되도록 했기 때문에 실제 얻어지는 출력 문자열은 "@nospace@-다"이다. 여기서 '@nospace@'는 뒤에 소개할 Coda 모듈을 위한 정보로서 나중에 자기 앞에 있는 공백을 지우는 역할을 하게 된다. 이러한 처리는 영어를

기준으로 하여 모든 형태소 사이에 공백을 넣어주는 KANT 시스템에서 명사와 조사, 어간과 어미를 구성하는 형태소 사이를 붙여쓰기 위해 도입되었다.

(morph-rule decl+p ((*) (+s "-다")))

이제 이 "제공-하-을 수 없" 문자열과 "@nospace@-다" 문자열이 어떻게 온전한 출력으로 연결될 지는 다음 단원에서 살펴보기로 하자.

본 연구에서는 이상 소개한 Generator 단계의 한국어 통사 구조 분석, 그리고 Morph 호출을 통한 한국어 문자열 생성을 가능하게 하기 위하여 19개의 문법 기호에 대한 73개 규칙 규모의 한국어 문법 규칙 정보, 115 어휘 유형 규모의 한국어 어휘 유형 정보, 166 어휘 규칙 규모의 한국어 어휘 규칙 정보를 구축하였다.

5. 한국어 다시 쓰기 규칙과 한국어 문장 생성

Coda 모듈은 최종 출력 문자열에 대한 다시 쓰기 규칙의 적용을 통하여 특수 기호나 음운 현상, 예외 현상 등의 처리를 수행한다.

앞 단원에서 예시한 서술어와 어미 부분에 대한 Generator 모듈의 출력은 아래와 같다. Generator는 서구 언어의 기준에 맞추어 만들어져 있기 때문에 서술어와 어미를 별개의 단어 사이로 간주하여 그 사이에 공백 문자를 삽입한다.

..... 제공-하-을 수 없 @nospace@-다

이제 위의 문자열에 적용될 수 있는 다시 쓰기 규칙들만 소개하면 다음과 같다.

(replace-string " @nospace@" "")
(replace-string "하-을" "할")
(replace-string "-" "")

이들 규칙의 첫 번째 인자에 매치되는 문자열을 두 번째 인자의 문자열로 고쳐 쓰는 과정을 반복할 경우 위의 문자열은 아래와 같이 고쳐진다.

제공-하-을 수 없[redacted]-다
⇒ 제공-[redacted] 수 없-다
⇒ 제공할 수 없-다
⇒ 제공할 수 없다
⇒ 제공할 수 없다

본 연구에서는 Coda 모듈의 이같은 처리를 위하여 68항 규모의 한국어 다시 쓰기 규칙 정보를 구축하였다.

6. 한국어 문장 생성 실험 결과

3, 4, 5절에서 소개한 한국어 언어 정보의 구축을 통하여 118 문장에 대한 중간 언어 표현 정보로 구성된 테스트 집합에서 106 문장에 대해 올바르며 완전한 한국어 문장 생성이 가능하게 되었다. 이러한 결과는 다른 언어에 대하여 수행된 유사한 연구의 결과와 비교할 때 대단히 우수한 성능으로 평가된다[9, 10].

테스트 집합 가운데 일부 문장의 생성 예 가운데 일부를 지면 관계상 중간 언어 표현이 아닌 영어 원시 문장과의 대조를 통해 제시하기로 한다.

The front hydraulic line provides fluid pressure to the hydraulic unit.
==> 앞쪽 유압선이 유압 장치로 유압을 제공한다.

The hydraulic pressure from the dual master cylinder is split diagonally between the right front/left rear wheel brake circuit and the left front/right rear wheel brake circuit.
==> 마스터 실린더 쌍으로부터의 유압은 전우륜-후좌륜 방향 바퀴 제동 회로와 전좌륜-후우륜 방향 바퀴 제동 회로 사이에 대각선 방향으로 나뉜다.

EBP (Electronic Brake Proportioning)
==> EBP(전자식 제동 조절)

All of the driving dynamics systems are actuated through the service brakes.
==> 모든 동적 운전 체제는 서비스 브레이크를 통하여 동작된다.

If at least one wheel begins to spin, the 4-ETS/ABS hydraulic unit applies brake pressure to the caliper of any spinning wheel.
==> 최소한 바퀴 하나가 회전하기 시작하면 4-ETS/ABS 방식 유압 장치가 회전하는 임의의 바퀴의 클리퍼스에 대한 브레이크 압력을 적용한다.

문장 생성에 실패하였거나 만족스러운 문장을 생성하지 못한 12 문장의 주요 실패 원인을 살펴보면 아래와 같다. 참고로 본 연구는 수행 과정상 시간적인 제약이 있었던 바, 대부분의 문제는 후속 연구를 통하여 해결이 가능할 것으로 본다.

(1) 동명사 등 품사의 전성이 일어나는 경우 : 'into thin-

king ...' 구절의 처리

In other words, the system fools the open differential into thinking that there is equal traction.

(2) 수사 문제 : '4 ...'로 번역됨. '4대의 ...'로 하고 싶으나 명사별로 단위가 달라짐.

4 speed sensors

(3) 삽입절의 처리 : 문장에 따라 다양한 속성의 빈 문장 성분이 생김.

The self-adjusting brake mechanism, which is like the W140, is located under the second seat row of the passenger compartment.

(4) 가주어 처리 : 문장 구조의 변경이 필요함

It is absolutely necessary that the driver operates the accelerator .....

(5) 일관성 없거나 잘못된 중간 언어 표현 : 처리 포기. 중간 언어 표현 변경을 요청함.

Low range is engaged.
// enganed가 부사로 잘못 표현되어 있음
A rough road is sensed.
// sensed가 수동태로 올바로 표현되어 있음.

7. 결 론

본 연구에서는 각종 한국어 언어 정보의 구축을 통하여 언어 독립적인 KANT 생성 엔진이 중간 언어 표현으로부터 한국어 문장을 생성해 낼 수 있도록 하였다.

이를 위하여 118 문장의 테스트 집합 처리를 위한 규모의 개념별 한국어 대응 규칙, 범주별 한국어 대응 규칙, 한국어 사전 및 템플리트 선언, 한국어 문법 규칙, 한국어 어휘 유형, 한국어 어휘 규칙, 한국어 다시 쓰기 규칙 등의 언어 정보를 구축하였으며 118 문장 규모의 테스트 집합에 대하여 106 문장에 대한 올바르며 완전한 한국어 문장 생성이 가능하였다.

분석기와 생성기를 분리시켜 다국어간 기계 번역을 용이하게 하는 한편 분석 엔진과 생성 엔진을 언어 중립적으로 구현하고 개별 언어 정보의 부가를 통하여 대상 언어를 확대하려는 KANT 연구진의 접근 방법은 특히 높이 평가할만하다. 이번 연구로 한국어가 KANT 목표 언어의 하나로 자리잡게 되었으며, 이는 영어 외에도 앞으로 추가될 모든 KANT 원시 언어에 대해 비록 제한된 도메인에서나마 한국어로의 기계 번역이 가능해졌다는 의미를 가진다.

본 연구에서 구축한 언어 정보가 제한된 테스트 집합만 대상으로 하였으므로 한국어 생성을 위한 본격적인 언어 정보 구축 작업이 차후 연구 과제로서 필요하다. 또한 분석기를 위한 한국어 언어 정보 구축 작업도 차후 필요한 연구 과제다.

참 고 문 헌

[1] Eric H. Nyberg, Teruko Mitamura, The KANT System : Fast, Accurate, High-Quality Translation in Practical Domains, in Proceedings of COLING '92, Nantes, France, July 1992.

[2] Eric H. Nyberg, Christine Kamprath, Teruko Mitamura, The KANT Translation System : From R & D to Large-Scale Deployment, LISA Newsletter, Vol.2:1, March, 1998, ISSN 1420-3693.

[3] Teruko Mitaruma, Eric H. Nyberg, 3rd, Jaime G. Carbonell, An Efficient Interlingua Translation System for Multi-lingual Document Production, Proceedings of Machine Translation Summit III, Washinton D.C., July, 1991.

[4] Jaime G. Carbonell, Masaru Tomita, "Knowledge based Machine Translation : The CMU Approach," in Nirenburg, S. (ed.), Machine Translation : Theoretical and Methodological Issues, New York : Cambridge University Press, 1987.

[5] Teruko Mitaruma, Eric H. Nyberg, Hierarchical Lexical Structure and Interpretive Mapping in Machine Translation, in Proceedings of COLING'92, Nantes, France, July 1992.

[6] 김영택, "어휘 기능 문법", in 김영택 외, 자연언어처리, 교학사, 1992.

[7] John R. Leavitt, KANT Mapper Specification, Technical Report, Carnegie mellon University - Center for Machine Translation, 1993.

[8] Masaru Tomita, Eric H. Nyberg, Generation Kit and Transformation Kit, Version 3.2, Users Manual, Technical Report, Carnegie mellon University - Center for Machine Translation, 1988.

[9] Dilep Zeynep Hakkani, Gökhan Tür, Teruko Mitamura, Eric H. Nyberg, and Kemal Oflazer, Issues in Generating Turkish from Interlingua, Technical Report, Carnegie mellon University - Center for Machine Translation, 1997.

[10] C. K. Turhan, An English to Turkish Machine Translation System Using Structural Mapping, In Proceedings of the 5th ACL Conference on Applied Natural Language processing, pp.320-323, 1997.

윤 덕 호

e-mail : dhyoon@eve.hannam.ac.kr

1985년 서울대 컴퓨터 공학과 졸업 (공학사)

1987년 서울대 계산통계학과 졸업 (이학석사)

1993년 서울대 컴퓨터 공학과 졸업 (공학박사)

1989년~현재 한남대학교 정보통신공학과 부교수

관심분야 : 자연언어처리, 웹 기반 정보처리, 멀티미디어