기계 번역을 위한 문법 기술 언어의 확장

심철민, 최승권, 여상화
한국전자통신연구원, 자연어처리연구부, 지식처리연구팀

An Extension of Grammar Writing Language for Machine Translation

Chul-Min Sim, Sung-kwon Choi, SangHwa Yuh
Knowledge Processing Team, Natural Language Processing Department, ETRI

요 약

변환 방식의 기계 번역 시스템에서 구조 변환은 번역의 품질을 결정하는 중요한 요소이다. 각 번역 시스템들은 이러한 구조 변환을 수행하기 위해 특별한 기법을 사용한다. 구조 변환을 수행하는 방안으로는 어휘 사전에 기술된 구조 변환 정보를 이용하는 방법, 변환 엔진에 언어 현상별 구조 변환 규칙을 프로그래밍하는 방법, 스크립트 언어를 이용하여 구조 변환 규칙을 기술하는 방법이 있다. 이 논문에서는 스크립트 형식의 범용 문법 기술 언어(Grammar Writing Language)를 제안한다. 이 논문에서 제안하는 문법 기술 언어는 규칙 기술을 용이하게 하기 위해 다양한 연산자와 기본 함수를 제공하며, 그 적용 대상에 따라 컴파일러 버전과 인터프리터 버전을 선택적으로 사용할 수 있다. 문법 기술 언어는 영한 기계 번역의 변환 모듈 뿐만 아니라 한영 변환 등의 트리 구조 변환을 요하는 다양한 응용 분야에 활용할 수 있다.

1. 서 론

변환 방식의 기계 번역 시스템에서 구조 변환은 번역의 품질을 결정하는 중요한 요소이다. 따라서 각 시스템별로 구조 변환을 수행하기 위해 특별한 기법을 사용한다. 구조 변환을 수행하는 방안으로는 어휘 사전에 구조 변환 정보를 함께 기술하여 변환 엔진에서 이 정보를 바탕으로 구조 변환을 수행하는 방법, 변환 엔진에 언어 현상에 대한 구조 변환 규칙을 프로그램으로 구현하는 방법, 그리고 별도의 스크립트 언어를 이용하여 각각의 언어 현상에 대한 구조 변환 규칙을 기술하고 이를 컴파일하여 변환 엔진에 통합하는 방법이 있다.

이 논문에서는 기존에 개발된 기계 번역용의 문법 기술 언어(Grammar Writing Language)를 범용으로 사용 가능하도록 확장한다[1, 2, 3]. 문법 기술 언어에서는 트리의 표현을 위한 다양한 연산자, 트리 변환을 위한 기본 함수들, 그리고 트리의 순위 조정 기능이 제공된다. 또한 문법 기술 언어로 기술된 트리 변환 규칙은 그 적용 대상에 따라 규칙의 순차적인 적용이나 직접적인 적용이 가능하다.

2. 문법 기술 언어의 요구 사항 및 문제점

기계 번역과 같이 트리 구조의 변환을 필요로 하는 응용 분야에서 트리 변환을 위해서는 다음과 같은 요구 사항을 만족해야만 한다.

(1) 선언적인 지식(Declarative Knowledge)과 절차적인 지식(Procedural Knowledge)의 분리
(2) 규칙의 가독성과 확장의 용이성
(3) 규칙 기술의 용이성
(4) 규칙의 모듈화
(5) 시스템 구성의 효율성
(6) 규칙에 대한 접근성
(7) 외부 자료 및 외부 모듈과의 연동성

기존의 문법 기술 언어는 이러한 요구 사항을 고려하여 전체적인 골격을 갖추고 있으나 실용적인 시스템에 적용하는데 다음과 같은 문제점을 안고 있었다.

(1) 트리의 자질에 대한 매크로 기능이 제공되지 않아 반복적으로 기술되는 변환 규칙이 존재한다.
(2) 규칙의 임의적인 접근 기능이 취약하다.

(3) 조건부 규칙의 기술이 취약하다.
(4) 임의 접근의 키워드로는 표제이만을 지원하므로 보다 다양한 규칙의 기술이 곤란하다.
(5) 문법 기술 언어 내부에서 프로그래밍 언어와의 연계가 불가능하다.
(6) 메모리 사용 및 실행 시간 지연이 많다.

3. 문법 기술 언어의 기술 형식

문법 기술 언어는 크게 3가지 요소로서 기술된다. 규칙의 대분류를 나타내는 GR(GRammar)과 GR을 구성하는 세부 규칙들의 그룹을 나타내는 PG(Part of Grammar) 그리고 PG를 구성하는 개별적인 변환 규칙인 TR(Transformational Rule)이 기본 요소들이다.

표 1. 문법 기술 언어의 기술 형식

GR 기술 형식	PG 기술 형식	TR 기술 형식

Grammar {
PG0
PG1
...
PGn
}
	Sample-PG {
	order h
	PG0
	...
	if (TR1) then
	{
	PGp1 TRpL'''
	} else {
	PGq1 TRqM'''
	}
	PGn
	}
		Sample-TR {
		_mode k
		(input tree pattern)
		_with {conditions}
		_var
		_tree ''' ;
		_feature ''' ,
		_action {
		statement
		}
		(output tree pattern)
		}

3.1 GR(GRammar)

GR은 하나 이상의 PG들로 구성된다. GR 내부의 PG들은 순차적으로 적용되며, GR은 트리 변환의 대분류로서 기술하는데 적합한 구조이다. GR을 기술하는 기본형태는 표 1과 같다.

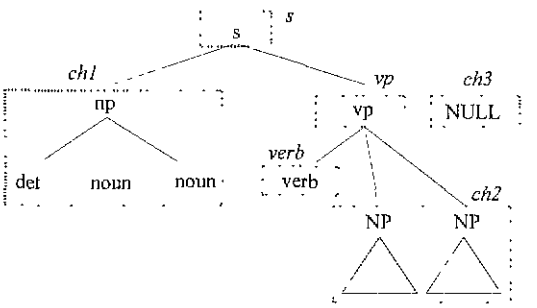
3.2 PG(Part of Grammar)

PG는 규칙을 모듈화하여 구분하는 부분이다. PG는 PG나 TR의 집합으로서 구성되며, 재귀적인 PG의 호출도 가능하다. PG 내의 구성 요소들의 수행 방법은 PG의 앞 부분에 명시하는 _order에 의해 결정된다. 즉, 최초 하나의 TR을 만족할 때까지 수행할 지, 모든 TR을 수행할 지 여부를 선택한다.

3.3 TR(Transformational Rule)

트리 변환을 위한 구체적인 규칙은 TR에서 기술된다. TR은 기술하고자 하는 규칙에 따라 전체의 트리에 대한 구조를 고려할 수도 있고, 부분적인 트리만을 고려할 수도 있다. _mode는 *input tree pattern*을 부분 트리에 반복적으로 적용할 지 여부를 정의한다.

(1) 트리 패턴 기술 : 입출력 트리의 패턴을 기술하는데 !(unique), ~(any), ?(one), &(optional)의 4가지 연산자가 이용되며, 서브 노드를 나타낼 경우에는 (와)를 이용하여 구조를 명시한다. 이를 이용하여 트리 패턴을 기술한 예는 다음과 같다.



(s! ~ch1 (vp! verb! ~ch2) ~ch3)
그림 1. 트리 구조의 표현과 해석

(2) 기본 함수 : 문법 기술 언어는 스크립트 형식으로 기술되므로 사전 접근 등의 복잡한 기능을 수행하기 위해 C 언어로 작성된 기본 함수를 제공한다. 기본 함수는 문법 개발자가 필요에 의해 정의하여 사용할 수 있다.
(3) 프로그래밍 언어 모듈과 연계 : 기본 함수를 정의하는 방법 외에도 모든 GR, PG및 TR에서 %%와 %% 사이에 외부 프로그램을 추가할 수 있다. 따라서 특정한 TR 내부에서 예외적인 처리가 필요하거나 중간 결과를 출력하는 등의 작업을 개발자가 임의로 기술할 수 있다. 또한 특정한 규칙을 보유한 모듈에 대해서만 별도의 C 함수를 정의하여 사용할 수도 있다.

4. 문법 기술 언어 실행기의 구현

이 논문에서 구현한 문법 기술 언어는 영한 기계 번역 시스템의 변환, 한영 기계 번역 시스템의 변환 모듈로 구현되어 있다[4] .

4.1 문법 기술 언어의 실행

이 논문에서 구현한 문법 기술 언어를 실제 시스템에 적용하기 위해서는 문법 기술 언어 외에도 여러 부가 모듈을 필요로 한다. 그림 2는 문법 기술 언어를 실행해 주는 실행 모듈의 구성도이다.

문법 기술 언어 실행기는 각 GR별로 입출력을 담당하는 문법 관리기, 문법 기술 언어로 기술된 변환 규칙, 기본 함수 및 저수준 함수 라이브러리, 문법 인터프리터, 메모리 관리기로 구성된다.

(1) 문법 관리기 : 문법 관리기는 메모리 초기화, 다수의 입력 트리에 대한 트리 변환 규칙 호출, 결과 트리의 순위 조정 기능을 수행한다.
(2) 트리 변환 규칙 : 문법 기술 언어로 기술된 규칙들은 컴파일러에 의해 컴파일된 C 프로그램 형태로 엔진에 통합된다. GR 함수를 시작으로 내부 함수를 순차적으로 수행

한다.

(3) 기본 함수 및 저수준 함수 라이브러리 : 앞 절에서 서술한 기본 함수 및 트리 변환 규칙을 구동하기 위해 필요한 저수준 함수가 구현되어 있는 모듈이다.

(4) 문법 인터프리터 : 임의 접근 규칙의 경우 탐색키에 대한 빈환 스크립트를 읽어 들여 이를 실시간에 해석하면서 규칙을 적용한다.

(5) 메모리 관리기 : 엔진의 처리 효율을 증가시키고 다수의 GR 모듈이 순차적으로 수행될 경우 부가적인 메모리 사용을 최소화하기 위해 메모리 관리자기 전체 메모리를 관리한다.

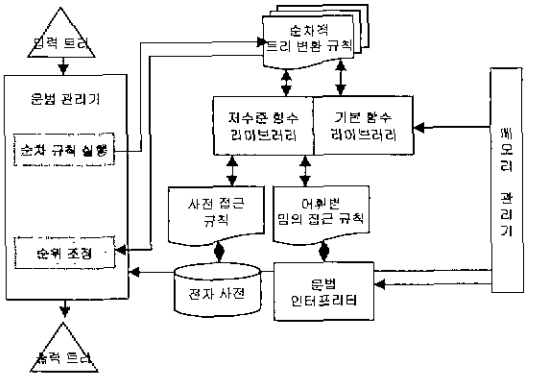


그림 2. 문법 기술 언어 실행 모듈의 구성도

4.2 순차 접근 및 임의 접근 규칙의 기술 예

문법 기술 언어의 컴파일 결과는 그 용도에 따라 C 프로그램이거나, 스크립트 명령문 형식이다. 기계 번역의 경우 트리 변환 규칙 부분은 규칙들이 순차적으로 적용되어야 하므로 C 프로그램 출력을 사용하며, 어휘별로 특별하게 기술한 번역 규칙의 경우는 임의로 접근 가능한 스크립트 명령문 출력을 사용한다. 표 2는 영한 기계 번역에서 기술된 순차 접근을 위한 변환 규칙과 임의 접근을 위한 변환 규칙의 예이다. 표 2와 같이 임의 접근 규칙의 경우 탐색키로는 표제어 뿐만 아니라 특정 지질값도 가능하다. 표 2의 예에서는 표제어 many에 대한 어휘 규칙과 의미표지(semantic marker) APF (인간의 감각적 속성)에 대한 변환 규칙을 보여준다.

표 2 순차 접근 및 임의 접근을 위한 규칙 기술 예

순차 접근 규칙

```
grammar {
    preprocessing-pg, ... }
preprocessing-pg {
    preprocess-type-1-tr, ... }
preprocess-type-1-tr {
    (np![1] (np'[2] (detp' (DET_' det')) ... )
    _with {
            det.root == [the],
            adj'root == [same];
            prep'root == [as], }
    _var _tree np,
    _action {
            np := _make_tree(np);
            np head      := [thing],
            ...
(np[1] np (pp (PREP_ PREP) (np[3] t1))) }
```

임의 접근 규칙

```
## many
many-pg {
    pre-sem-mk-comn-tr, ... }
pre-sem-mk-comn-tr {
    (pre*! ~ch1 comn! ~ch2)
    _with {
            pre*:sem_mk1 <- [CA],
            comn s_case == [subj],
            comn.head == [many];
            comn.head_cat << [noun pron], }
    _action {
            comn'trn_k_lex1  = [|*많은 사람*|];
            comn.trn_k_gcode1 := [NN00001];
            comn trn_k_pos := [NOUN]; }
    (pre* ch1 comn ch2) }
## sem_mk|APF
color-pg {
    cira-in-color-tr; ... }
cira-in-color-tr {
    (comn! Cira! ~ch )
    _with {
            comn:head_cat == [noun],
            comn:sem_mk == [CAH];
            cira:prep_lex == [in];
            cira:sem_mk == [APF]; }
    _action {
            cira josa_lex  = [|*옷을 입은*|],
            cira josa_gcode  = [NE00005], }
    (comn  cira ch) }
```

5. 결 론

이 논문에서는 스크립트 형식의 범용 문법 기술 언어(Grammar Writing Language)를 제안하였다. 이 논문에서 제안하는 문법 기술 언어는 변환 규칙 기술을 용이하게 하기 위해 다양한 연산자와 기본 함수를 제공하며, 그 적용 대상에 따라 컴파일러 버전과 인터프리터 버전을 선택적으로 사용할 수 있다. 문법 기술 언어는 영한 기계 번역의 변환 모듈 뿐만 아니라 한영 변환 등의 트리 구조 변환을 요하는 다양한 응용 분야에 활용할 수 있다.

참 고 문 헌

[1] 박수근, 목구조 변형을 기반으로 한 문법기술언어의 확장 및 개선, 1990년도 정보과학회 가을 학술발표논문집, 1990.

[2] 권철중, 변환 방식의 기계 번역을 위한 문법 기술 언어의 설계 및 구현, 한국과학기술원 전산학과, 1989.

[3] 손덕진, 문법 기술 언어 GWL-II의 설계 및 구현, 제2회 기계번역 Workshop 발표논문집, 1989.

[4] 심철민, 에서로-웹/EK : 영한 웹 문서 번역 시스템, 제9회 한글 및 한국어 정보처리 학술발표논문집, 1997.

[5] 강원석, 영한 기계 번역 시스템의 문법 개발 시스템과 지원 도구 설계 및 구현, 1989년도 정보과학회 가을 학술발표논문집, 1989

[6] Jun-ichi NAKAMURA, Grammar Writing System (GRADE) of Mu-Machine Translation project and its characteristics, COLING-84, 1984.