영한자동번역에서의 두단계 영어 전산문법

최승권*†
한국전자통신연구원

Sung-Kwon Choi. 2000. Two-level English Computational Grammar in English-to-Korean Translation. *Language and Information 4.1*, 97–109. Application systems of natural language processing such as machine translation system must deal with actual texts including the full range of linguistic phenomena. But it seems to be impossible that the existing grammar covers completely such actual texts because they include disruptive factors such as long sentences, unexpected sentence patterns and erroneous input to obstruct well-formed analysis of a sentence. In order to solve analysis failure due to the disruptive factors or incorrect selection of correct parse tree among forest parse trees, this paper proposes two-level computational grammar which consists of a constraint-based grammar and an error-tollerant grammar. The constraint-based computational grammar is the grammar that gives us the well-formed analysis of English texts. The error-tolerant computational grammar is the grammar that reconstructs a comprehensible whole sentence structure with partially successful parse trees within failed parsing results. **(Electronics and Telecommunications Research Institute)**

1. 서론

자동번역에서의 영어 구문 분석을 위한 기존의 언어학 문법들은 대부분 문법적인 문장을 대상으로 문법규칙을 작성하여 왔다. 하지만 우리가 실생활에서 흔히 접하는 실제 영어 문장들은 단순하고 문법적인 문장들 이외에도 길고 복잡할 뿐만 아니라 비문법적인 문장들이 포함되어 있으며 문법규칙으로 설명하기 어려운 패턴들이 들어 있기도 하다.

비문법적이어서 영어 분석을 제대로 못하는 경우는 제외하고라도 문법적인 문장임에도 파싱시 잘못 분석되는 경우는 영한 자동번역에서 자주 접할 수 있으며 결과적으로는 번역품질을 떨어뜨리는 원인이 되어 왔다.

이러한 점에서 본 논문에서는 영한 자동번역에서 번역의 품질을 높일 수 있기 위하여 영어의 문법적인 문장을 올바른 한국어 문장으로 번역할 수 있도록 하고 비문법적이거나 분석 실패한 문장이라도 가능한 한 한국어 문장으로 번역할 수 있도록 하는 영한 자동번역에서의 영어 구문분석을 위한 두단계 영어전산 문법을 제안하고자 한다.

* 대전광역시 유성구 가정동 161번지 한국전자통신연구원 지식처리연구팀 305-350,
E-mail:choisk@etri.re.kr. Fax:82-42-860-4889

† 본 논문에 대해 심사해 주신 두 분의 익명의 심사위원에게 감사의 말씀을 드립니다. 본 논문의 두 단계 전산문법은 번역실험을 해 본 특정 영한자동번역시스템에 의존적일 수 있으며 차트(Chart parser)의 개념을 기초로 하여 기술된 논문입니다. 본 논문의 영한 자동번역을 위한 두 단계 전산문법은 아직도 개선 중에 있으며 본 논문에서의 부족한 부분은 모두 저자의 책임임을 밝힙니다.

본 논문에서 소개하고자 하는 영어 전산 문법은 일명 두 단계 전산문법이라고 불려질 수 있는데 왜냐하면 단계적으로 적용되기 때문이다. 첫번째 전산문법인 제약기반 전산문법은 정형적이고 문법적인 문장에 대해 올바른 분석을 제공하는 문법을 말하며 두번째 전산문법인 오류허용문법은 제약기반 전산문법에서 문장 전체에 대해 분석 실패한 경우에 제약기반 전산문법에서 올바르게 구문 분석한 구단위의 분석 결과들을 가지고 재조립하여 가능한 한 한국어로 자동번역하도록 돕는 문법을 말한다.

본 논문은 다음과 같이 구성된다. 2장에서는 웹사이트에 나타난 영어 문장들의 언어적인 현상을 살펴보고자 한다. 3장에서는 정형적이며 문법적인 영어 문장을 처리할 수 있는 제약기반 문법을 소개하고자 한다. 4장에서는 제약기반 전산문법에서 문장 전체에 대해 분석 실패한 경우에 제약기반 전산문법에서 올바르게 구문 분석한 구단위의 분석 결과들을 가지고 재조립 하여 가능한 한 한국어로 자동 번역하도록 돕는 오류허용문법에 대해 기 술하고자 한다. 5장에서는 두단계 전산문법에 의해 영어 구문분석을수행 하고 있는 영한 웹 자동번역 시스템에 의한 영어 구문분석의 실험결과를 제시할 것이다.

2. 영어 웹 문자의 언어 현상

2.1 언어현상적인 특징

웹에 등장하는 영어 텍스트들의 언어 현상적 특징을 정리하면 다음과 같 다.

(1) 가. 긴 고유 명사구의 반번한 출현

예) Worldata's International Lists and Databases

나. 도메인이나 도메인 그룹에 따른 특수 구문의 존재

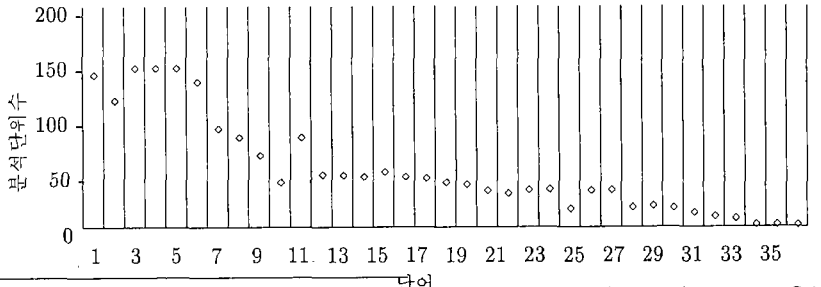
예) 날씨: 요일 night 월 일, 년

다. 다양한 기호의 존재

예) I am also happy to post incremental versions (i.e. if you have an hour, do a few changes, and then someone else can improve on that...).

2.2 통계적 특징

여러 웹사이트 중에서 컴퓨터 관련 웹사이트를 대상으로 수집된 2141문장[1]의 웹 문서를 단어 길이별로 그 분포를 살펴보았을 때 다음과 같았다:



1. 컴퓨터 관련 웹사이트로부터 추출된 2141문장은 http://dir.yahoo.com/Science/ Computer-Science/ 에 있는 50개 범주들로부터 임의로 추출한 문장들의 총합이다.

[그림1]은 컴퓨터관련 영어 웹 문서의 문장은 1단어로부터 8단어로 이루어진 문장이 전체 문장에서 50%이상을 차지하며 기타 20단어 이상으로 구성되는 문장들도 자주 나타나지만 한 문장이 36단어 이상으로 만들어지는 문장은 매우 적다는 것을 알 수 있다. 비록 이러한 분석이 컴퓨터 분야의 웹 문서에 대해서만 분석되었지만 다른 범주의웹 문서에서도 비슷한 결과를 낼 것이라 생각된다.

3. 제약기반 전산문법

제약기반 constraint-based 전산문법은 문법적인 영어 문장의 올바른 분석 및 구조적 중의성 해결을 담당하는 전산문법이다.

3.1 구문분석 문법 Formalism

제약기반 전산문법의 Formalism은 영어가 형상구조 configurational language라는 가정 하에 구성성분구조를 기술할 수 있는 문맥자유문법의 형식이 사용되었으며 구조적 중의성을 해결하기 위해 문맥자유문법에 제약조건이 부여되어 있는 형식이다. 이러한 점에서 제약기반 전산문법은 확장된 문맥자유문법 Augmented CFG으로써 기존의 문맥자유문법에 조건부와 실행부라는 자질 제약부를 가지는 구조이다.(최기선 (1992)) 제약기반 전산문법에서의 규칙의 형식을 나타내면 다음과 같다:

(2) 제약기반 전산문법의 규칙 형식
0: 부모노드 → 자식노드들
{ 조건부 }
{ 실행부 }

조건부와 실행부에서는 연산자들에 의해 자질들이 적용되게 되는데 조건부와 실행부에서 사용하는 연산자들을 소개하면 다음과 같다:

【도표1】 제약기반 전산문법의 연산자들

조건부		실행부	
연산자	의미	연산자	의미
==	equal	:=	assign
!=	not equal		
**	boolean intersection		

LFG (Kaplan and Bresnan (1982))나 HPSG (Pollard and Sag (1994))와 같은 기존의 통합문법 Unification Grammar처럼 제약기반 전산문법도 통합 unification이 기본적인 정보조작이다.

3.2 구문노드 및 필요정보

제약기반 전산문법의 적용력을 향상시킬 수 있는 방법은 규칙들간의 충돌이 없으며 다양한 언어현상을 설명할 수 있는 대용량의 제약기반 전산규칙을 구축하는 것이다. 이를 위해 우선적으로 고려되어야 하는 것은 전산문법의 기술의 편리성을 높여야 한다는 것이다. 이를 위해 본 제약기반 전산문법은 해당 구구조 노드의 정보를 그 노드의 머리범주로부터 일관되게 복사하는 정보복사 방법을 사용하였다. 이는 기존의 머

리어주도구구조문법인 HPSG (Pollard and Sag (1994))에서의 머리자질이동과 유사한 방법이다. 본 논문에서의 구문노드와 그것의 머리범주간의 정보이동은 다음의 도표와 같이 나타낼 수 있다(지면관계상 상세 정보는 생략하였다):

【도표 2】 제약기반 전산문법의 정보이동표

구문노드		통사정보		의미정보
노드명	설명	머리정보	부가정보	머리정보
ADNP	부사화명사구	HEAD	cnode,lnode,rnode	HEAD
ADP	부사구	HEAD	cnode,lnode,rnode conj,nega,wh	HEAD
AP	형용사구	HEAD	cnode,lnode,rnode conj,wh	HEAD
CMPCL	비교절	HEAD	cnode,lnode, rnode,conj	HEAD
CMPR	비교사	HEAD	cnode,lnode,rnode	HEAD
CNCL	양보절	HEAD	cnode,lnode,rnode	HEAD
DETP	관사구	HEAD	cnode,lnode,rnode	HEAD
INFP	부정사구	HEAD	cnode,lnode, rnode(missing)	HEAD
INGP	ING구	HEAD	cnode,lnode, rnode,nega	HEAD
NP	명사구	HEAD	cnode,lnode, rnode,nega,wh cu,det_ lex conj,pp_ attach	HEAD
PAPP	과거분사구	HEAD	cnode,lnode, rnode,missing	HEAD
PP	전치사구	HEAD	cnode,lnode, rnode,wh	HEAD
QSBCL	인용절	HEAD	cnode,lnode, rnode	HEAD
S	문장	HEAD	cnode,lnode, rnode,conj, nega,pseudo,subj	HEAD,tense, aspect,modal, voice,mood
SBCL	종속절	HEAD	cnode,lnode, rnode,conj	HEAD
SBCLM	성분생략종속절	HEAD	cnode,lnode, rnode,missing	HEAD
SM	성분생략문	HEAD	cnode,lnode, rnode,missing	HEAD
TAGP	부가의문구	HEAD	cnode,lnode,rnode	HEAD
VP	동사구	HEAD	cnode,lnode, rnode,pseudo	HEAD,tense, aspect,modal voice,mood
VPM	성분생략동사구	HEAD	cnode,lnode,rnode, missing	HEAD
WHADP	WH-부사구	HEAD	cnode,lnode,rnode, wh	HEAD
WHAP	WH-형용사구	HEAD	cnode,lnode,rnode	HEAD
WHCL	WH-절	HEAD	cnode,lnode,rnode	HEAD
WHINFP	WH-부정사구	HEAD	cnode,lnode,rnode	HEAD
WHNP	WH-명사구	HEAD	cnode,lnode,rnode, wh	HEAD

속성	
aspect(상)	number(수)
cnode(핵심노드이름)	person(인칭)
conj(병렬여부)	pp_attach
cu(단어의 숙어여부)	pseudo(가주어/ 가목적어 표시)
det_lex(한정사의 어휘)	rnode(오른쪽 노드종류)
form(비교급,최상급, 활용형태	root(표제어)
gender(성)	sem(어휘의미)
lnode(왼쪽노드)	subj(의미상주어여부)
missing(생략된 요소)	tense(시제)
modal(양상조동사)	type(동사유형)
mood(화법)	voice(태)
nega(부정여부)	wh(wh여부)

단어노드		통사정보		의미정보
노드명	설명	머리정보	부가 정보	머리정보
ADJ	형용사	root,type,form		sem
ADV	부사	root,type,form		sem
AUX	조동사	root,type,form		tense
CONJ	접속사	root,type		
DET	관사	root,type		
NOUN	명사	root,type,form		sem
PREP	전치사	root		
PRON	대명사	root,type,gender, number,person		
VERB	동사	root,type,form		sem

위의 도표에서 각 구문노드 Non-terminal nodes는 그것의 머리범주노드로부터 통사적 정보와 의미적 정보를 복사받고(위의 도표에서는 HEAD라는 이름으로 언급하였음) 머리범주노드와는 상관없이 해당 노드의 정보만을 기술하기 위해서 사용하는 노드는 cnode, lnode, rnode와 같이 상수값으로 해당 노드에 기술된다. 그리고 단어노드 terminal nodes는 상위로 보낼 정보인 머리정보로 구성된다. 위의 도표에 따라 문법기술자는 규칙 기술시에 각 노드에 정해진 정보만을 일관되게 기술할 수 있기 때문에 규칙기술을 위한 지적 부담을 덜 수 있고 규칙 충돌을 피해갈 수 있다.

이상의 제약기반 전산문법을 위한 규칙기술 방법에 의해 문장단위의 규칙과 동사구 단위의 규칙을 기술하는 방법을 기술하면 다음과 같다 (지면 관계상 상세 정보는 생략하였다):

(3) 가. 문장에 관한 제약기반 문법규칙

```
0: S→ NP VP
{ }
{/* SYNTACTIC HEAD */
S:type := VP:type;
/* SEMANTIC HEAD */
S:sem := VP:sem;
S:tense := VP:tense;
/* STATISTIC */
```

S:score := [1.0];
/* STRUCTURE INFO */
S:lnode := [phr];
S:rnode := VP:rnode;}

나. 타동사구에 관한 제약기반 문법규칙
0: VP → VERB NP
{VERB_:v_ type ** [T1]; }
{/* SYNTACTIC HEAD */
VP:type := VERB:type;
/* SEMANTIC HEAD */
VP:sem := VERV:sem;
VP:tense := VERB:tense;
/* STATISTIC */
VP:score :=[1.0];
/* STRUCTURE INFO */
VP:lnode := VERB:lnode;
VP:rnode := [phr];}

설명: 제약기반 전산문법은 인덱스 0으로써 기술되며 문장 S와 동사구 VP는 각각 하위 머리범주인 동사구와 동사로부터 통사적, 의미적 머리정보인 동사타입 type, 의미 sem, 시제 tense등을 전달받고 확률점수 [1.0]을 가지게 된다. 또한 상위범주들은 하위 머리범주로부터 좌우접속정보 lnode, rnode를 전달받게 된다.(지면상 상세 정보는 생략하였다)

위의 예는 문장과 타동사구의 문법규칙 정보구조가 해당 상위노드 하위노드이름을 제외하고 정보가 동일하게 적용된다는 것을 보여준다.

3.3 구조적 중의성 해결

구조적 중의성이란 동일한 문장에 대해 한 개 이상의 규칙이 적용됨으로써 한 개이상의 분석결과가 나오는 것을 말한다. 구조적 중의성 해결은 전산 언어학에서 가장 해결하기 어려우며 꼭 해결해야 할 문제이다. 구조적 중의성을 해결하기 위해 여러 가지전산학적인 해결책이 소개되기는 하였지만(Franz (1992)) 아직도 해결하여야 할 것이 많은 실정이다.

본 논문에서도 완전한 해결책은 아니지만 영어 구문분석에서 구조적 중의성을 해결하기 위해 사용한 언어학적 장치를 기술고자 한다.

3.3.1 좌우접속 구조정보에 의한 해결. 각 구문구조에서의 구성성분이 좌우로 접속된 구조정보를 가지게 함으로써 잘못 구성되는 구문구조의 중의성을 해결할 수 있다. 좌우로 접속되는 구조정보는 다음과 같은 도표로 나타낼 수 있다:

【도표 3】 좌우접속 구조정보 표

속성	값
lnode(좌접속 구조정보)	nil(없음),phr(구),cla(절)
rnode(우접속 구조정보)	nil(없음),phr(구),cla(절)

위에서 좌우접속 구조정보의 값으로 세 종류인 nil, phr, cla가 사용되었는데 nil은

좌우접속구조가 없을 때 부여되는 값이고, phr은 좌우접속구조가 동사구를 포함하지 않을 때 부여하는 값이며 cla는 동사구를 포함하는 절이나 문장이 좌우접속구조로 올 때 부여하는 값이다.

위의 좌우접속 구조정보 표에 의해 아래와 같은 동사구 규칙은 통합조작에 의해 중의성이 없어지게 된다.

(4) 좌우접속 구조정보에 의한 구조중의성 해결 예

가) NP→ NP INFP {NP:rnode :=[cla]}

나) VP→ VERB NP {NP:rnode ** [nil phr cla]}

다) VP→ VERB NP INFP {NP:rnode ** [nil phr]}

설명) (4가)의 상위노드 명사구의 우접속 정보 rnode은 하위노드 INFP 때문에 값으로 절 cla을 가지는데 (4나)의 동사구에서는 명사구의 우접속 자질로 절을 허용하지만 (4다)의 동사구에서는 명사구의 우접속 자질로 절을 허용하지 않기 때문에 (4가)의 부정사구를 가지는 명사구는 세번째와 같은 동사구에서 명사구로 실현될 수 없다.

3.3.2 동사유형정보에 의한 해결. 동사유형은 동사를 분류하기 위해 동사의 필수성분의 수와 형태를 기반으로 만든 것으로 제약에 사용함으로써 구조적 중의성 해결에 활용할 수 있다.

영어 동사를 형식과 성분수로 유형을 나누어보면 다음의 도표와 같다:

【도표 4】 동사유형정보 표

	0 (없음)	1 (명사구)	2 (부정사구)	3 (to부정사구)	4 (동명사구)	5 (절)	6 (형용사구)	7 (분사구문)	8 (부사구/전치사구
I(1형식)	I0								
L(2형식)		I1		L3	L4	L5	L6		L8
T(3형식)		T1	T2	T3	T4	T5			
D(4형식)		D1				D5			
X(5형식)		X1	X2	X3	X4		X6	X7	X8

위의 동사유형정보 표에 의해 동일한 형태소 태깅의 문장이 동사의 유형에 따라 구조분석이 달라 질 수 있다는 것을 다음의 예문을 통해 알 수 있다:

(5) 가) I call you to meet him.

s(np(I) vp(vp(call you) infp(to meet him))). (call 동사타입 T1)

나) I meet you to meet him.

s(np(I) vp(want you infp(to meet him))). (want 동사타입 X3)

설명) (5가)와 (5나)의 문장은 동사 call, want만 다를 뿐 동일한 형태소 분석결과를 가진다. (5가)에서는 call동사의 동사타입이 명사구를 요구하 는 타동사이기 때문에 call you가 동사구로 먼저 묶였고 (5다)에서는 동사 want가 목적보어를 요구하는 동사타입이기 때문에 want you to meet him이 동사구로 묶인 것이다.

3.3.3 사전의 전치사 필수성분 정보에 의한 해결. 전치사 접속에 따른 구조적 중의성은 pp-attachment로써 널리 알려진 구조적 중의성이다.(Whitelock and Kilby (1995)) 이러한 pp-attachment를 해결하기 위해 본 연구에서는 사전에 등록된 전치사 필수성분 정보를 이용하였다. 다음의 규칙들이 pp-attachment의 해결을 보인다:

(6)가) NP → NP PP {NP:pp_ attach := PP:preplex}

나) VP → VERB NP {VERB:arg2_ prep != NP:pp_ attach}

다) VP → VERB NP PP

설명) (6가)의 상위 명사구 NP의 pp_ attach의 값은 전치사구 PP의 전치사 어휘 preplex로터 부여받는다. (6나)는 동사 사전에 들어 있는 동사의 두 번째 성분의 전치사와 명사구의 전치사가 일치하지 않아야 한다는 것으로 필수적인전치사구를 요구하는 동사는 (6다)의 규칙에 적용을 받을 것이다.

4. 오류허용 전산문법

오류허용 전산문법이란 제약기반 전산문법에서 문장 전체에 대해 구문분석이 오류 혹은 실패한 경우에 차트 파싱(Chart parsing)의 기술에 근거하여 제약기반 전산문법에서 올바르게 구문 분석한 구구조 분석 결과들을 가지고 문장 구조를 재구성하여 가능한 한 한국어로 자동번역하도록 돕는 문법을 말한다.

4.1 구문분석(parsing) 실패 원인 분석

다양한 언어현상을 가지는 영어 웹 문서를 구문분석(parsing)할 때 다양한 구문분석 실패가 발생하는데 그 원인을 요약하면 다음과 같다:

(7) 구문분석(parsing)의 실패 원인들

가. 수많은 파싱 결과 중에 정확한 분석 결과를 찾지 못하는 경우.

나. 영어 웹문장에 대응할 구문분석규칙이 없을 경우.

다. 구조적 중의성을 해결하지 못하였을 경우.

라. 장문의 영어 웹문장에 대해 처리하지 못하였을 경우.

마. 사전 엔트리의 정보를 잘못 기술하였을 경우.

바. 비문법적인 문장의 입력시 대응을 하지 못할 경우.

이상의 구문분석 실패 원인에 대처하기 위해 다음 절서부터는 영한 자동 번역에서의 오류허용 전산문법에 대해 기술하고자 한다.

4.2 오류허용 전산문법의 Formalism

오류허용 전산문법의 Formalism는 제약기반 전산문법의 Formalism와 동일한 형식이나 제약기반 전산문법이 제약조건을 가지고 있는 반면 오류 허용 전산문법은 그러한 제약조건을 가지지 않는다. 오류허용 전산문법의 구성형식은 다음과 같다:

(8) 오류허용 전산문법의 규칙 형식

1: 부모노드 → 자식노드들
{ 없음 }
{ 실행부 }

오류허용 전산문법의 실행부에서도 자질이 사용되는데 이 자질들은 제약기반 전산문법의 자질과 동일하다. (2)에서 기술된 제약기반 전산문법에서의 동사구에 대한 오류허용 전산문법에서의 기술은 다음과 같다:

(9) 가. 타동사구에 관한 오류허용 문법규칙 1: VP → VERB NP

```
{ }
{/* SYNTACTIC HEAD */
VP:type := VERB:type;
/* SEMANTIC HEAD */
VP:sem := VERV:sem;
VP:tense := VERB:tense;
/* STATISTIC */
VP:score := [1.0];
/* STRUCTURE INFO */
VP:lnode := VERB:lnode;
VP:rnode := [phr];}
```

설명: 오류허용 전산문법은 인덱스 1로써 기술되며 실행부에서의 기술은 제약기반 전산문법과 동일하나 조건부에 대한 제약이 없다는 것이 제약 기반 전산문법과 틀리다.(지면상 상세 정보는 생략하였다)

위에서 소개된 타동사구 규칙에 따르면 철자 오류라든가 완전한 문장을 이루지 못하였거나 성분이 필요이상 많은 비문법적인 문장도 다음과 같이 구문분석을 수행 후 올바른 번역을 할 수 있었다.

(10) 가) I gave a book. ⇒ 나는 책을 주었다.

나) He sleeps a room ⇒ 나는 방에(에서) 잔다.

(10가) 문장은 give가 여격동사임에도 불구하고 오류허용 규칙에 의해 묶여 번역이 성공적으로 수행된 결과이며 (10나) 문장은 sleep이 자동사임에도 불구하고 타동사 구문처럼 묶인 후 격을 가지지 않는 명사구에 기본적 으로 붙는 한국어 격조사 '에(에서)'가 붙어 성공적으로 번역된 문장이다.

4.3 구문노드 및 필요정보

오류허용 전산문법에서 사용하는 정보는 제약기반 전산문법에서 사용하는 정보와 동일한 것을 사용하였다. 그 이유는 (1) 제약기반 전산문법에서 사용한 정보를 오류허용 전산문법에서 사용하더라도 문법기술에는 전혀 문제가 없었으며 (2) 문법작성자가 제약기반 전산문법에서 사용한 정보와는 다른 정보를 오류허용문법에서 사용하였을 때 지적부담이 많아 대량의 분석규칙을 작성하는 데 어려움이 있기 때문이다. 오류허용 전산문법에서의 구구조 노드의 정보는 제약기반 전산문법에서의 정보복사와 마찬가지로 머리범주의 정보가 상위노드의 정보로 일관되게 복사되는 정보복사 방법을 사용하였다. 오류허용 전산문법에서의 구문노드와 그것의 머리범주간의 정보이동은 다음의 도표와 같다(지면관계상 모든 정보는 기술하지 못하였다)

【도표 5】 오류허용 전산문법의 정보이동표

구문노드		통사정보		의미정보
노드명	설명	머리정보	부가정보	머리정보
CMPCL	비교절	HEAD	cnode,lnode, rnode,conj	HEAD
CMPR	비교사	HEAD	cnode,lnode, rnode	HEAD
CNCL	양보절	HEAD	cnode,lnode, rnode	HEAD
INFP	부정사구	HEAD	cnode,lnode, rnode(missing)	HEAD
INGP	ING구	HEAD	cnode,lnode, rnode,nega	HEAD
PAPP	과거분사구	HEAD	cnode,lnode, rnode,missing	HEAD
QSBCL	인용절	HEAD	cnode,lnode, rnode	HEAD
S	문장	HEAD	cnode,lnode, rnode,conj nega,pseudo, subj	HEAD,tense, aspect,modal ,voice,mood
SBCL	종속절	HEAD	cnode,lnode, rnode,conj	HEAD
SNCL	가정법절	HEAD	cnode,lnode, rnode	HEAD
VP	동사구	HEAD	cnode,lnode, rnode,pseudo	HEAD,tense, aspect,modal modal,voice,mood
WHAP	WH-형용사구	HEAD	cnode,lnode, rnode	HEAD
WHCL	WH-절	HEAD	cnode,lnode, rnode	HEAD
WHINFP	WH-부정사구	HEAD	cnode,lnode, rnode	HEAD

단어노드		통사정보		의미정보
노드명	설명	머리정보	부가정보	머리정보
CONJ	접속사	root,type		
VERB	동사	root,type,form		sem

속성	
aspect(상)	number(수)
cnode(핵심노드이름)	person(인칭)
conj(병렬여부)	pp_ attach
lnode(왼쪽노드)	subj(의미상주어여부)
modal(양상조동사)	type(동사유형)
mood(화법)	voice(태)
nega(부정여부)	wh(wh여부)

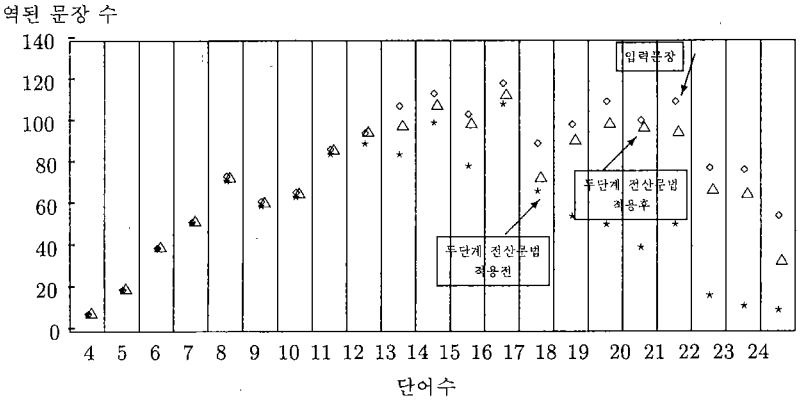
5. 적용 실험

본 절에서는 두 단계 영어 전산문법이 들어있는 영한 웹 자동번역기에 의해 인터넷에서 영어 웹문서를 한국어로 자동번역한 번역결과를 토대로 하여 두 단계 영어 전산문법의 실험 결과를 기술하고자 한다. 영한 자동번역기에서 사용한 두 단계 전산문법의 규칙 크기는 다음과 같았다:(1999년 12월 1일 현재 크기임) [2]

【도표 6】 번역규칙의 크기

규칙		
영어구문 분석규칙	제약기반 규칙	447개
	오류허용 규칙	81개

두단계 영어 전산문법의 효과를 객관적으로 검증하기 위해 영어 웹문서 중에서 임의의 문장 1719문장을 대상으로 두 단계 영어 전산문법이 없을 때와 있을 때를 비교하였을 때 다음의 그림과 같은 차이를 보였다:



【그림 2】 두 단계 영어 전산문법에 의한 번역품질 개선 효과

그림2는 입력문장에 대해 두 단계 전산문법이 없이 자동번역할 때와 두 단계 전산문법을 가지고 자동번역을 할 때 번역시스템이 11단어 이상의 문장에서부터 더 좋은 번역품질을 나타내는 것을 보인다.[3]

2. 오류허용 전산문법에서의 규칙의 수는 제약기반 전산문법의 전체 규칙의 수에서 상위노드가 ADNP(부사화 명사구), AP(형용사구), ADP(부사구), NP(명사구), PP(전 치사구), TAGP(부가의문문구), WHADP(Wh-부사구), WHNP(Wh-명사구) 인 규칙들의 수를 뺀 나머지 수이다. 위와 같은 상위노드들이 오류허용 전산문법에서 배제된 이유는 동사와 관련한 노드들이 차트파싱시 대량의 구조적 모호성으로 인해 구문분석실패를 만드는 반면 위와 같은 노드들은 구조적 모호성이 대부분 제약기반 전산문법에서 해소되기 때문이다.

3. 여기서 평가의 기준은 임의의 문장에 대해 자동번역한 결과가 다음의 평가점수표에서 등급 1 이상의 점수를 받은 문장수의 합을 의미한다.

본 논문에서 소개된 영한 웹 번역기에 의해 자동번역된 결과를 선보이면 다음과 같다:

【그림 3】 99년12월6일자 "USA TODAY"웹사이트의 번역결과

6. 맺음말

지금까지 이 논문에서는 영한자동번역에서의 영어 구문분석을 위한 두 단계 영어 전산문법을 기술하였었다. 두 단계 영어 전산문법은 정형적이고 문법적인 문장에 대해 올바른 분석을 제공하는 제약기반 전산문법과 제약기반 전산문법에서 문장 전체에 대해 분석 실패한 경우에 제약기반 전산문법에서 올바르게 구문 분석한 구단위의 분석 결과들을 가지고 재조립하여 가능한 한 한국어로 자동번역하도록 돕는 오류허용문법이었다.

본 논문에서 기술된 두 단계 영어 전산문법은 아직도 수정 보완되어야 할 부분이 많지만 현재까지 파악된 두 단계 영어 전산문법의 단점을 보완하기 위해 앞으로 고려

등급	의미
4(Perfect)	번역된 문장의미가 완전함.
3(Good)	번역된 문장의미가 거의 완전함.
2(OK)	번역된 문장의미가 몇번 읽은 후에야 이해됨.
1(Poor)	번역된 문장의미가 많이 읽은 후에야 이해됨
0(Fail)	번역된 문장의미가 파악되지 않음

하고자 하는 작업은 (1) 제약기반 전산문법에 패턴 개념을 도입하여 영어의 구조 분석 시 구조적 모호성을 더욱 낮추고자 하며 (2) 오류허용 전산문법에 구문노드간의 경쟁적 확률정보를 도입하여 더욱 옳게 성공한 문장 구조를 찾아내고자 준비하고 있다.

참고문헌

최기선, 외. 1992. **영한기계번역시스템:문법개발지원환경 및 해석문법개발**. 한국과학기술원, 과학기술처.

Franz, A. 1992. *Ambiguity Resolution in Language Processing*. CMT, Carnegie Mellon University.

Kaplan, Ronald and Joan Bresnan. 1982. Lexical functional grammar:a formal system for grammatical representation. In John Bresnan, editor, *The Mental Representation of Grammatical Relations*. Cambridge:MIT, pages 173–281.

Pollard, Carl and Ivan Sag. 1994. *Head-Driven Phrase Structure Grammar .Studies in Contemporary Linguistics*. The University of Chicago and London, Chicago and London.

Whitelock, P and K Kilby. 1995. *Linguistics and Computational Techniques in Machine Translation Design*. UCL.

접수일자: 1999년 12월 15일
게재결정: 2000년 3월 31일