Implementation of Presence and Absence of Blocking Effects: A Categorial Grammar Approach to Chinese and Korean*

Yong-hun Lee

Dept. of English Lang. & Lit. Chungnam Nat'l Univ. & Hannam Univ.
220 Gung-dong, Daejeon 305-764, Korea
ylee@cnu.ac.kr

Abstract. Among the languages that allow long-distance reflexives, some languages have blocking effects, whereas others don't. The goal of this paper is to provide computational algorithms that can handle presence and absence of blocking effects of long-distance reflexives. We will examine the blocking effects in Chinese and Korea and develop computational algorithms for handling blocking effects in those two languages. The algorithms will be developed by incorporating Chierchia's Binding Theory into Steedman's Combinatory Categorial Grammar (CCG). Through the analyses and implementations, this paper illustrates how blocking effects can be implemented computationally.

Keywords: long-distance reflexive, blocking effects, Chinese, Korean, CCG

1. Introduction

In some languages, long-distance reflexives are allowed in addition to their sentence-bound counterparts. Among the languages that allow long-distance reflexives, some languages have blocking effects, but others don't. Chinese is one language that has blocking effects and the sentence (1) demonstrates an example of blocking effects (Cole, et al., 2000:14).[1]

(1) Blocking Effect in Chinese

Zhangsan_i	renwei	wo_j	zhidao	Wangwu_k	xihauan	$\text{ziji}_{*i/*j/k}$
Zhangsan	think	I	know	Wangwu	like	self

'Zhangsan thinks I know Wangwu likes self.'

Here, the reflexive *ziji* cannot refer to *Zhangsan* because *wo* blocks co-reference between *ziji* and *Zhangsan*. Let's compare this sentence with (2). (2) is the Korean counterpart of sentence (1).[2]

[1] Cole et al. (2000) included state-of-art introduction to long-distance reflexives, and various approaches to long-distance reflexives are tried by Pica (1987), Manzini & Wexler (1987), Battistella (1989), Katada (1991), Haung & Tang (1991), Cole & Sung (1994), etc. Discussions on Chinese long-distance reflexives are contained in Huang (1982), Hung (1984), Tang (1989), Haung & Tang (1991), among others.
[2] We have similar phenomenon in the following sentence (Moon, 1996:15).

(i)

John_i-un	[*nay_j-ka*	*$\text{caki}_{i/*j}$-lul*	*ttayli-ess-ta-ko*]	*syangkakha-n-ta*.
John.TOP	I.NOM	self.ACC	beat.PAST.DECL.COMP	think.PRES.DECL

'John thinks that I beat him.'

Here, the reflexive *caki* can refers to *John* across the pronoun *nay*. If Korean had blocking effects, this phenomenon would be impossible since the pronoun *nay* blocks the co-referential relations between *caki* and *John*.

(2) No Blocking Effect in Korean

Chelsoo$_i$*-nun* [*nay*$_j$*-ka* [*Younghee*$_k$*-ka* *caki*$_{i/*j/k}$*-lul* *co.aha-n-ta-ko*]
Chelsoo.TOP I.NOM Younghee.NOM self.ACC like.PRES.DECL.COMP
a(*l*)*-n-ta-ko*] *sayngkakha-n-ta*.
know.PRES.DECL.COMP think.PRES.DECL
'Chelsoo thinks that I know that Younghee likes him/herself.'

As the co-reference relations in (2) indicate, *caki* CAN refer to *Chelsoo*, though *nay* is located between them. Therefore, we can say that there is no blocking effect in Korean long-distance reflexives.

The goal of this paper is to provide computational algorithms that can handle presence and absence of blocking effects. The algorithms will be developed by combining Steedman (1996, 2000)'s Combinatory Categorial Grammar (CCG) and Chierchia's Binding Theory, which will be called a CCG-like system.

This paper is organized as follows. In Section 2, Categorial Grammar and Chierchia (1988)'s Binding Theory will be introduced. Section 3 introduces a CCG-like system and demonstrates how blocking effects can be handled in the CCG-like system. Section 4 provides computational algorithms for presence and absences of the blocking effects, and Section 5 summarizes this paper.

2. Binding Theory in Categorial Grammar

Categorial Grammar was first introduced by Ajdukiewicz (1935) and later modified and advanced by Bar-Hillel (1953), Curry & Feys (1958), and Lambek (1958). In this framework, we have two basic categories *n* and *s*, and other categories come from the combinations of these two categories. All the syntactic phenomena are described and analyzed by the functor-argument relations of the constituents.

Steedman (1996, 2000) extended previous studies in Categorial Grammar and developed Combinatorial Categorial Grammar (CCG). The most important characteristic of his system is that predicate-arguments relations are projected by the combinatory rules of syntax, and other operations are based on these predicate-arguments relations (Steedman, 2000:38). The most fundamental combinatory rule is functional application, which is delineated in (3). Here, *f* is the semantic interpretation of the functor category, and *a* is that of the argument.

(3) Functional Application (Steedman, 1996:13)

a. $X / Y : f \quad Y : a \quad \rightarrow \quad X : f a$ (>)

b. $Y : a \quad X \backslash Y : f \quad \rightarrow \quad X : f a$ (<)

Chierchia applied Categorial Grammar to explain Binding phenomena in English, and he described syntactic constraints of reflexives and pronominals as follows (Chierichia, 1988:134).

(4) Binding in Categorial Grammar

a. A reflexive must be bound to an F-commanding argument in its minimal NP or S domain.

b. A non-reflexive pronoun must not be co-indexed with anything in its minimal NP or S domain.

where F-command is simply c-command at function-argument structure.

Agreement in number and gender must hold between pronouns and their antecedents, in order to pronouns can refer to their antecedents. The constraint for checking agreement is stated in (5a).

FT(*n*) in (5b) has three information: *n* is the index of the NP, *gndr* is gender, and *nmbr* is number (Chierchia, 1988:132).

(5) Agreement-Checking Algorithm

a. FT(*n*)≈FT(*m*): The features associated with *n* are non-distinct from those associated with *m*.

b. $$\mathrm{FT}(n) = \begin{pmatrix} n \\ \mathrm{gndr} \\ \mathrm{nmbr} \end{pmatrix}$$

For example, the FTs of three different NPs *John*, *himself*, and *her* can be stated as follows.

(6)

$$\underset{\mathrm{John}_1}{\mathrm{FT}(1) = \begin{pmatrix} 1 \\ \mathrm{male} \\ 3 \end{pmatrix}} \qquad \underset{\mathrm{himself}_2}{\mathrm{FT}(2) = \begin{pmatrix} 2 \\ \mathrm{male} \\ 3 \end{pmatrix}} \qquad \underset{\mathrm{her}_3}{\mathrm{FT}(3) = \begin{pmatrix} 3 \\ \mathrm{female} \\ 3 \end{pmatrix}}$$

Chierchia introduced resolution algorithms for pronouns in English based on the combinatorics, and they are enumerated in (7). Here and throughout, integers will be used as names for the categories mentioned in the rules.

(7) Chierchia's Algorithms (1988:138-9)

a. $\underset{0}{\mathrm{TV}} + \underset{1}{\mathrm{NP}} \Rightarrow \underset{2}{\mathrm{IV}}$

(i)	$\mathrm{LPS}(0) \cap \mathrm{LPS}(1) = \varnothing$	non-coreference
(ii)	$\mathrm{SLASH}(2) \cap (\mathrm{LPS}(1) \cup \mathrm{LPS}(2)) = \varnothing$	crossover
(iii)	$\mathrm{SLASH}(2) = \mathrm{SLASH}(0) \cup \mathrm{SLASH}(1)$	slash-percolation
(iv)	$\mathrm{LPS}(2) = \mathrm{LPS}(0) \cup \mathrm{LPS}(1)$	LPS-percolation

b. $\underset{0}{\mathrm{S/NP}_n} + \underset{1}{\mathrm{NP}_n} \Rightarrow \underset{2}{\mathrm{S}}$

(i)	$\mathrm{LPS}(2) = \varnothing$	A-opacity boundary
(ii)	$\mathrm{SLASH}(2) \cap (\mathrm{LPS}(1) \cup \mathrm{LPS}(2)) = \varnothing$	crossover
(iii)	$\mathrm{SLASH}(2) = \mathrm{SLASH}(0) \cup \mathrm{SLASH}(1)$	slash-percolation
(iv)	$\underset{+\mathrm{refl}}{n} \notin \mathrm{LPS}(0) \cup \mathrm{LPS}(1)$	reflexive

c. $\underset{0}{\mathrm{IV/IV}} + \underset{1}{\mathrm{IV}} \Rightarrow \underset{2}{\mathrm{IV}}$

(i)	$\mathrm{LPS}(2)=\mathrm{LPS}(0)$	A-opacity boundary
(ii)	$\mathrm{SLASH}(2) \cap (\mathrm{LPS}(1) \cup \mathrm{LPS}(2)) = \varnothing$	crossover
(iii)	$\mathrm{SLASH}(2) = \mathrm{SLASH}(0) \cup \mathrm{SLASH}(1)$	slash-percolation
(iv)	$\underset{+\mathrm{refl}}{\mathrm{n}} \notin \mathrm{LPS}(1)$	reflexive

d. Reflexives

(i) $\underset{n\,[+\mathrm{refl}] \in \mathrm{LPS}}{A} \Rightarrow \underset{n \notin \mathrm{LPS}}{A}$

(ii) conditions: (a) A = IV, TV (b) $\mathrm{FT}(A) \approx \mathrm{FT}(n)$

(iii) translation: $\lambda x_n\,[A'(x_n)]$

3. Analyzing Blocking Effects with Category

3.1. A CCG-like System

The system that this paper develops is a CCG-like system, which has been introduced in Lee (2002a, 2002b, 2003a, 2003b). It is basically an incorporation of Chierchia's ideas into Steedman's Combinatory Categorial Grammar (CCG). This system is similar to Steedman's system in that surface combinatorics triggers other operations, especially reflexive resolution algorithms in this paper. It is different from Steedman's in that it uses attribute-value ordered pairs (**avop**) in (8) to describe syntactic dependencies of constituents. The six attributes are explained in (9).

(8) Structure of Attribute-Value Ordered Pair
<PHON,CAT,(AGR),TRANS,NPS,(SLASH)>

(9) Six Attributes

a. PHON
- (i) phonological/morphological form
- (ii) concatenates a word to a stream of words

b. CAT
- (i) has categorial information
- (ii) such as S, NP, S\NP, and so on

c. AGR
- (i) agreement feature
- (ii) index, type, gender, and number

d. TRANS
- (i) semantic interpretation
- (ii) based on Montagovian semantics

e. NPS (NP Index Store)
- (i) something like a Cooper-storage
- (ii) stores indexes of NP

f. SLASH
- (i) similar to that of HPSG
- (ii) deals with crossover phenomena

The functional application on the CAT values triggers operations on TRANS and NPS values, and all the reflexives are resolved by these operations. NPS is similar to Chierchia's LPS, but different in (i) that NPS stores the indexes for other types of NPs in addition to pronouns and (ii) that NPS is not local. That is, the indexes stored in NPS can be percolated up beyond the minimal S domain. AGR and SLASH are parenthesised in (8), because the values for these two attributes will be omitted from the actual representations.

The reflexive resolution algorithms in the CCG-like system can be described as follows. Each constituent in the input sentence is combined with the others by functional applications in (3). After this combination, if there is [+refl] in the NPS store, it triggers reflexive resolution algorithms. The resolution algorithms perform some operations on TRANS and NPS values, and all the reflexives are resolved with their antecedents.

3.2. Blocking Effects in the CCG-like System

Now, let's see how presence and absence of blocking effects can be handled in the CCG-like system. First, in order to process long-distance reflexives, Chierchia's algorithms in (7) must be revised slightly. Among the conditions that are enumerated in (7), the conditions that prohibit long-distance reflexives are (7bi) and (7biv). Both conditions say that reflexives must be resolved in the minimal S domain, and that [+refl] indexes cannot be percolated up beyond the minimal S boundary. In order to handle sentences in (1) and (2), these conditions have to be deleted. (10) and (11) are the algorithms after this revision. (10) is for Chinese and (11) is for Korean.

(10) Revised Chierchia's Algorithms for Chinese

a. $\underset{0}{(S\backslash NP_n)/NP_n} + \underset{1}{NP} \Rightarrow \underset{2}{S\backslash NP}$

(i) $NPS(0) \cap NPS(1) = \varnothing$ — non-coreference
(ii) $SLASH(2) \cap (NPS(1) \cup NPS(2)) = \varnothing$ — crossover
(iii) $NPS(2) = NPS(0) \cup NPS(1)$ — NPS-percolation
(iv) $SLASH(2) = SLASH(0) \cup SLASH(1)$ — slash-percolation

b. $\underset{0}{NP_n} + \underset{1}{S\backslash NP_n} \Rightarrow \underset{2}{S}$

(i) $SLASH(2) \cap (NPS(1) \cup NPS(2)) = \varnothing$ — crossover
(ii) $NPS(2) = NPS(0) \cup NPS(1)$ — NPS-percolation
(iii) $SLASH(2) = SLASH(0) \cup SLASH(1)$ — slash-percolation

c. Reflexives

(i) $\underset{n\,[+\text{refl}] \in NPS}{A} \Rightarrow \underset{n \notin NPS}{A}$
(ii) conditions: (a) $A = S\backslash NP$ (b) $FT(A) \approx FT(n)$
(iii) translation: $\lambda x_n\,[A'(x_n)]$

(11) Revised Chierchia's Algorithms for Korean

a. $\underset{0}{NP} + \underset{1}{(S\backslash NP_n)\backslash NP_n} \Rightarrow \underset{2}{S\backslash NP}$

(i) $NPS(0) \cap NPS(1) = \varnothing$ — non-coreference
(ii) $SLASH(2) \cap (NPS(1) \cup NPS(2)) = \varnothing$ — crossover
(iii) $NPS(2) = NPS(0) \cup NPS(1)$ — NPS-percolation
(iv) $SLASH(2) = SLASH(0) \cup SLASH(1)$ — slash-percolation

b. $\underset{0}{NP_n} + \underset{1}{S\backslash NP_n} \Rightarrow \underset{2}{S}$

(i) $SLASH(2) \cap (NPS(1) \cup NPS(2)) = \varnothing$ — crossover
(ii) $NPS(2) = NPS(0) \cup NPS(1)$ — NPS-percolation
(iii) $SLASH(2) = SLASH(0) \cup SLASH(1)$ — slash-percolation

c. Reflexives

(i) $\underset{n\,[+\text{refl}] \in NPS}{A} \Rightarrow \underset{n \notin NPS}{A}$
(ii) conditions: (a) $A = S\backslash NP$ (b) $FT(A) \approx FT(n)$
(iii) translation: $\lambda x_n\,[A'(x_n)]$

(7c) is deleted here, since we will not use this category combinatorics. Note that the conditions (7bi) and (7biv) are deleted in (10) and (11). Also, note that NP follows $(S\backslash NP_n)/NP$ in (10a) but NP precedes $(S\backslash NP_n)\backslash NP$ in (11a). This is just a language-specific property.

Revised algorithms in (10) and (11) are for handling long-distance reflexives. Now, it's time to develop algorithms for handling blocking effects. For this purpose, this paper adopts Reflexive-Antecedent Pairing Algorithm in (12) that are developed in Lee (2002b, 2003a).

(12) Reflexive-Antecedent Pairing Algorithm
For the reflexive *r* and all the potential antecedents *a*, make a pair '*r* = *a*' if category of *a* is NP.

These processes start from the closest antecedent from the reflexive *r* and continue until all the possible antecedents are exhausted. In addition to this algorithm, to handle differences between

Chinese and Korean, we have algorithms in (13) and (14) for blocking effects.

(13) Blocking Effect Algorithm for Chinese
When we meet a reflexive-antecedent pair whose agreement feature is not compatible, delete all the reflexive-antecedent pairs after this pair.

(14) Blocking Effect Algorithm for Korean
When we meet a reflexive-antecedent pair whose agreement feature is not compatible, delete only this reflexive-antecedent pair and comparison processes continue.

Now, let's take (1) and (2) again, and see how these algorithms work.
First, according to the algorithm in (12), Reflexive-Antecedent Pairs for sentence (1) and (2) can be calculated as in (15) and (16).

(15) Reflexive-Antecedent Pairs for (1) (Step I)
a. ziji = Wangwu
b. ziji = wo
c. ziji = Zhangsan

(16) Reflexive-Antecedent Pairs for (2) (Step I)
a. caki-lul = Younghee-ka
b. caki-lul = nay-ka
c. caki-lul = Chelsoo-nun

Then, compatibility between the reflexive and its antecedents are checked from (a) to (c). Because the reflexive and the antecedents in (15b) and (16b) are not compatible, (15b) and (16b) are deleted as in (17) and (18).

(17) Reflexive-Antecedent Pairs for (1) (Step II)
a. ziji = Wangwu
b. ~~ziji = wo~~
c. ziji = Zhangsan

(18) Reflexive-Antecedent Pairs for (2) (Step II)
a. caki-lul = Younghee-ka
b. ~~caki-lul = nay-ka~~
c. caki-lul = Chelsoo-nun

Because we meet a Reflexive-Antecedent Pair whose agreement feature is not compatible, Blocking Effect Algorithms in (13) and (14) are applied, and we reach the final results as in (19) and (20).

(19) Reflexive-Antecedent Pairs for (1) (Final)
a. ziji = Wangwu
b. ~~ziji = wo~~
c. ~~ziji = Zhangsan~~

(20) Reflexive-Antecedent Pairs for (2) (Final)
a. caki-lul = Younghee-ka
b. ~~caki-lul = nay-ka~~
c. caki-lul = Chelsoo-nun

Following the Reflexive-antecedent Pairs in (19), (21) is analyzed as in (22), where (21) is the same sentence as (1) except that bracketed numbers are added by superscription. These numbers refer to morphological/phonological forms of each lexical item. Note that the algorithm in (10c) is applied to the S\$NP_3$ node. By (10ci), 4_{+refl} is deleted from the NPS. (10cii) is satisfied because the current category is S\NP. By (10cii), the semantic interpretation of the S\$NP_3$ node is changed from *like'*(x_4) into λx_4[*like'*(x_4, x_4)]. After this node meets *Wangwu*, *ziji* is resolved only with *Wangwu*, as *like'*(*w*,*w*) indicates.

(21) [1]Zhangsan$_i$ [2]renwei [3]wo$_j$ [4]zhidao [5]Wangwu$_k$ [6]xihauan [7]ziji$_{*i/*j/k}$
Zhangsan think I know Wangwu like self
'Zhangsan thinks I know Wangwu likes self.'

(22)

<[1]+…+[7],S,think'(*z*,^know'(I,^like'(*w*,*w*))),NPS:1_{+name},2_{+pron},3_{+name}>

<[1],NP_1,*z*,NPS:1_{+name}><[2]+…+[7],S\NP_1,think'(^know'(I,?like'(*w*,*w*))),NPS:2_{+pron},3_{+name}>

<[2],(S\NP_1)/S,think',NPS:1><[3]+…+[7],S,know'(I,^like'(*w*,*w*)),NPS:2_{+pron},3_{+name}>

<[3],NP_2,I,NPS:2_{+pron}><[4]+…+[7],S/NP_2,know'(^like'(*w*,*w*)),NPS:3_{+name}>

<[4],(S\NP_2)/S,know',NPS:2><[5]+…+[7],S,like'(*w*,*w*),NPS:3_{+name}>

<[5],NP_3,*w*,NPS:3_{+name}> <[6]+[7],S\NP_3,λx_4[like'(x_4,x_4)],NPS:3>

<[6]+[7],S\NP_3,like'(x_4),NPS:3,4_{+refl}>

<[6],(S\NP_3)/NP,like',NPS:3><[7],NP_4,x_4,NPS:4_{+refl}>

Likewise, (23) can be analyzed into either (24) or (25), based on the Reflexive-Antecedent Pairs in (20). Here also, (23) is the same sentence as (2) except that bracketed numbers are added by superscription for morphological/phonological forms.

(23) [1]*Chelsoo*$_i$*-nun* [[2]*nay*$_j$*-ka* [[3]*Younghee*$_k$*-ka* [4]*caki*$_{i/*j/k}$*-lul* [5]*co.aha-n-ta-*[6]*ko*]
Chelsoo.TOP I.NOM Younghee.NOM self.ACC like.PRES.DECL.COMP
[7]*a*(*l*)*-n-ta-*[8]*ko*] [9]*sayngkakha-n-ta*.
know.PRES.DECL.COMP think.PRES.DECL
'Chelsoo thinks that I know that Younghee likes him/herself.'

(24)

<[1]+…+[9],S,think'(*c*,^know'(I,^like'(*y*,*y*))),NPS:1_{+name},2_{+pron},3_{+name}>

<[1],NP_1,*c*,NPS:1_{+name}><[2]+…+[9],S\NP_1,think'(^know'(I,^like'(*y*,*y*))),NPS:2_{+pron},3_{+name}>

<[2]+…+[8],S',know'(I,^like'(*y*,*y*)),NPS:2_{+pron},3_{+name}><[9],(S\NP_1)\S',think',NPS:1>

<[2]+…+[7],S,know'(I,^like'(*y*,*y*)),NPS:2_{+pron},3_{+name}><[8],S'\S,$\lambda\phi[\phi]$,NPS:∅>

<[2],NP_2,I,NPS:2_{+pron}><[3]+…+[7],S\NP_2,know'(^like'(*y*,*y*)),NPS:3_{+name}>

<[3]+…+[6],S',like'(*y*,*y*),NPS:3_{+name}><[7],(S\NP_2)\S',know',NPS:2>

<[3]+…+[5],S,like'(*y*,*y*),NPS:3_{+name}><[6],S'\S,$\lambda\phi[\phi]$,NPS: ∅>

<[3],NP_3,*y*,NPS:3_{+name}> <[4]+[5],S\NP_3,λx_4[like'(x_4,x_4)],NPS:3>

<[4]+[5],S\NP_3,like'(x_4),NPS:3,4_{+refl}>

<[4],NP_4,x_4,NPS:4_{+refl}><[5],($S\backslash NP_3$)/NP,like’,NPS:3>

(25) <[1]+…+[9],S,think’(*c*,^know’(I,^like’(*y*,*c*))),NPS:1_{+name},2_{+pron},3_{+name}>

<[1]+…+[9],S,λx_4[think’(*c*,^know’(I,^like’(*y*,x_4)))](*c*),NPS:1_{+name},2_{+pron},3_{+name}>

<[1]+…+[9],S,think’(*c*,^know’(I,^like’(*y*,x_4))),NPS:1_{+name},2_{+pron},3_{+name},4_{+refl}>

<[1],NP_1,*c*,NPS:1_{+name}><[2]+…+[9],$S\backslash NP_1$,think’(^know’(I,^like’(*y*,x_4))),NPS:2_{+pron},3_{+name},4_{+refl}>

<[2]+…+[8],S’,know’(I,^like’(*y*,x_4)),NPS:2_{+pron},3_{+name},4_{+refl}><[9],($S\backslash NP_1$)\S’,think’,NPS:1>

<[2]+…+[7],S,know’(I,^like’(*y*,x_4)),NPS:2_{+pron},3_{+name},4_{+refl}><[8],S’\S,$\lambda\phi[\phi]$,NPS:∅>

<[2],NP_2,I,NPS:2_{+pron}><[3]+…+[7],$S\backslash NP_2$,know’(^like’(*y*,x_4)),NPS:3_{+name},4_{+refl}>

<[3]+…+[6],S,like’(*y*,x_4),NPS:3_{+name},4_{+refl}><[7],($S\backslash NP_2$)\S’,know’,NPS:2>

<[3]+…+[5],S,like’(*y*,x_4),NPS:3_{+name},4_{+refl}><[6],S’\S,$\lambda\phi[\phi]$,NPS:∅>

<[3],NP_3,*y*,NPS:3_{+name}><[4]+[5],$S\backslash NP_3$,like’(x_4),NPS:3,4_{+refl}>

<[4],NP_4,x_4,NPS:4_{+refl}><[5],($S\backslash NP_3$)/NP,like’,NPS:3>

In (24), note that the algorithm in (11c) is applied to the $S\backslash NP_3$ node, and *caki* is resolved only with *Younghee*. In (25), the algorithm in (11c) is applied to the $S\backslash NP_1$ node, and *caki* is resolved only with *Chelsoo*.

4. Implementation of Blocking Effects

Now, it’s time to develop algorithms for handling blocking effects. Those algorithms can be divided into two steps. The first one is to calculate reflexive-antecedent pairs such as those in (15) or (16), and the second step is to filter out the reflexive-antecedent pairs whose agreement feature is not compatible. (26) is the algorithm for calculating reflexive-antecedent pairs. That is, (26) is an implementation of idea in (12).

(26) Algorithm for Reflexive-Antecedent Pairing

```
function set reflexive_antecedent_pairing(int i, int j, int k)
var
    string rPHON, aPHON;
    set RAPairs;
begin
    RAPairs := {};
    r := k;
    rPHON := PHON(Lex[NPs[r]]);
    for a from j to i step ?1 do
        begin
            aPHON := PHON(Lex[NPs[a]]);
            RAPairs := RAPairs + ‘rPHON=aPHON’;
        end;
    return(RAPairs);
end;
```

Here, Lex[] is an array where every lexical item is stored, and NPs[] is an array where all the NPs are stored. That is, NPs[] corresponds to NPS in the analyses in (22), (24), and (25). k refers to the position of the reflexive in NPs[]. Searching domain for possible antecedents is from NPs[i] to NPs[j]. The algorithm searches this domain backwards, i.e., from NPs[j] to NPs[i], and makes a Reflexive-Antecedent Pair 'rPHON = aPHON'. Here, rPHON is the phonological/morphological form of the reflexive, and aPHON refers to that of the possible antecedents. Then, the algorithm adds this pair to RAPairs, where all the reflexive-antecedent pairs are stored. After all the NPs are exhausted, the algorithm returns RAPairs.

The next step is to check agreement feature of each pair in RAPairs, to rule out the pairs where the reflexive and a possible antecedent are not compatible. (27) and (28) are those algorithms, where (27) is the implementation of (13) and (28) is that of (14).

(27) Algorithm for Blocking Effects (Chinese)

```
function set Chinese_blocking_effects (set ra)
var
    boolean flag;
    string RAString, RARefl, RAAnte;
    set RAPairs, RASet;
begin
    flag := true;
    RAPairs := ra;
    RASet := {};
    while flag = true and RAPairs is not exhausted do
        begin
            get one RAString from RAPairs;
            RARefl := reflexive form of RAString;
            RAAnte := antecedent form of RAString;
            if (FT(RARefl)?FT(RAAnte)) flag := false;
            if (flag = true) then RASet := RESet + RAString;
        end;
    return(RASet);
end;
```

(28) Algorithm for Blocking Effects (Korean)

```
function set Korean_blocking_effects (set ra)
var
    boolean flag;
    string RAString, RARefl, RAAnte;
    set RAPairs, RASet;
begin
    flag := true;
    RAPairs := ra;
    RASet := {};
    while RAPairs is not exhausted do
        begin
            get one RAString from RAPairs;
            RARefl := reflexive form of RAString;
            RAAnte := antecedent form of RAString;
            if (FT(RARefl)?FT(RAAnte)) flag := false;
            if (flag = true) then RASet := RESet + RAString;
            if (flag = false) then flag := true;
        end;
    return(RASet);
end;
```

If the agreement feature between the reflexive and its possible antecedent is not compatible, *flag* is set to be false. When *flag* is false, different actions are taken between Chinese and Korean. In Chinese, as the algorithm in (27) demonstrates, we get out of the while loop, and returns the reflexive-antecedent pairs whose agreement feature is compatible up to now. In Korean, as the algorithm in (28) shows, we filter out only that reflexive-antecedent pair, and continue comparison processes until all the pairs in RAPairs are exhausted. Along with these algorithms, we can handle presence and absence of blocking effects in Chinese and Korean effectively.

5. Conclusion

In this paper, we have developed algorithms for handling blocking effects of long-distance reflexives. Chinese has blocking effects, but Korean doesn't have. We found that long-distance reflexives in these languages can be analyzed efficiently in the CCG-like system and that the presence and absence of blocking effects can be implemented by some operations on the reflexive-antecedent pairs.

References

Ajdukiewicz, K. 1935. Die Syntaktische Konnexitat. *Studia Philisophica*, 1, 1-27.

Bar-Hillel, Y. 1953. A Quasi-arithmetical Notation for Syntactic Description. *Language*, 29, 47-58.

Battistella, E. 1989. Chinese Reflexivization: A Movement to INFL Approach. *Linguistics*, 27, 987-1012.

Chierchia, G. 1988. Aspects of a Categorial Theory of Binding. In R. Oehrle et al. eds., *Categorial Grammars and Natural Language Structures*, 125-51. Dordrecht: R. Reidel.

Cole, P., G. Hermon, and J. Huang. 2000. Introduction: Long-Distance Reflexives, the State of the Art. *Syntax and Semantics*, 33, xiii-xlvii.

Cole, P. and L. Sung. 1994. Head Movement and Long-distance Reflexives. *Linguistic Inquiry* 21, 1-22.

Curry, H. and R. Feys. 1958. *Combinatory Logic*. Vol. 1. Amsterdam : North Holland.

Huang. J. 1982. *Logical Relations in Chinese and the Theory of Grammar*. Ph.D. thesis. MIT.

________. 1982. On the Distribution and Reference of Empty Pronouns. *Linguistic Inquiry*, 15, 531-574.

Huang, J. and L. Liu. 2000. Logophoricity, Attitude, and *Ziji* at the Interface. In Cole, P., G. Hermon, and J. Huang. 2000. *Syntax and Semantics* 33, 141-195.

Huang, J. and J. Tang. 1991. On the Local Nature of the Long-distance Reflexive in Chinese. In Koster, J. and E. Reuland, eds., *Long-distance Anaphora*, 263-282. Cambridge: Cambridge University Press.

Katada, F. 1991. The LF Representation of Anaphors. *Linguistic Inquiry*, 22, 287-313.

Lee, Y. 2002a. Implementation of Long-distance Reflexives in Korean. *Proceedings of the 16th Pacific Asia Conference on Language, Information, and Computation* (*PACLIC-16*), 296-307.

______. 2002b. Resolving A Korean Reflexive Cakicasin: A Categorial Grammar Approach. *Studies in Linguistics Science*, 32(1), 55-77. UIUC, Urbana.

______. 2003a. Bound vs. Long-distance Reflexives in Korean: A Categorial Grammar Approach. In G. Iverson and S. Ahn, eds., *Explorations in Korean Language and Linguistics*, pp. 423-41. Seoul : Hankook Publishing Co.

______. 2003b. *Anaphora Resolution Algorithms with Category*: *English and Korean*. Ph.D. thesis. University of Illinois at Urbana-Champaign.

Lambek, J. 1958. The Mathematics of Sentence Structure. *American Mathematical Monthly*, 65, 154-170.

Manzini, M and K. Wexler. 1987. Parameters, Binding Theory, and Learnability. *Linguistic Inquiry*, 18, 414-444.

Moon, Seung-Chul. 1996. *An Optimality Approach to Long Distance Anaphors*. Ph.D. thesis. University of Washington. Seoul: Thaehaksa.

Pica, P. 1987. On the Nature of Reflexivization Cycle. *Proceedings of NELS*, 17, 483-499. GLSA, University of Massachusetts at Amherst.

Steedman, M. 1996. *Surface Structure and Interpretation*. Cambridge, MA: MIT Press.

__________. 2000. *The Syntactic Process*. Cambridge, MA: MIT Press.

Tang, C. 1989. Chinese Reflexives. *Natural Language and Linguistic Theory*, 7, 93-121.