Lambda Abstraction and Scrambling*

Yoon-kyoung Joh
(Hankuk University of Foreign Studies)

Joh, Yoon-kyoung. 2011. Lambda Abstraction and Scrambling. *The Linguistic Association of Korea Journal. 19*(1). 1-22. This paper compares two versions of lambda abstraction proposed by Heim and Kratzer (1998) and Bittner (1994). Even though the former is considered to be more standard than the latter in the literature, many data have proven that the latter is more useful than the former. This paper claims that the latter should be adopted in the grammar in that Bittner's rule not only helps us to treat scrambling efficiently but also fundamentally explains why it occurs and when it can and cannot occur. The fact that arguments can be arranged in any order at syntax is directly motivated by the semantic rule that allows any variable with an index to be lambda-abstracted freely. Since Bittner's rule is empirically attested by one of the prevalent linguistic phenomena called scrambling, the rather liberated power of Bittner's rule that might give rise to some concerns can be justified. Bittner's rule removes unnecessary complications in the grammar while placing unavoidable complexity in the right place.

Key Words: scrambling, lambda abstraction, function application, predicate modification, indices

1. Introduction

In the literature, various semantic rules have been proposed especially to account for the compositional aspect of interpretations. Lambda abstraction is one of the rules that plays a crucial role in semantics. The lambda abstraction rule proposed by Heim and Kratzer (1998) is very often employed when such a

* I appreciate three anonymous reviewers for their priceless comments and insightful suggestions. Of course, all remaining errors are mine.

rule is necessary almost as a standard form. However, several alternative forms of the rule exist, showing a different degree of restrictedness. This paper will review Bittner's (1994) lambda abstraction rule in particular and discuss the advantages it has over Heim and Kratzer's.

I will primarily provide the prevalent linguistic phenomenon of scrambling as supporting evidence and conclude that Bittner's (1994) lambda abstraction rule is more desirable than Heim and Kratzer's (1998) rule. In (1), we can see that the order of the two arguments can be shifted. Looking at such data, I will claim that the flexibility of syntactic constituents is basically attributed to the corresponding flexibility in semantics that is expressed by Bittner's rule. That is, under the system proposed by Bittner (1994), we can get the answers not only to the question how we can handle the cross-linguistically attested phenomenon of scrambling efficiently but also to the more fundamental questions why we have scrambling and when we can and cannot have scrambling. The bottom line of my point will be that scrambling is not a deviated form but a natural structure that is semantically motivated in a principled manner.

(1.a)	Minho-ka	chayk-ul	ilk-ess-ta.
	Minho-Nom	book-Acc	read-Pst-Dec
	'Minho read a book.'		
(1.b)	Chayk-ul	Minho-ka	ilk-ess-ta.
	book-Acc	Minho-Nom	read-Pst-Dec
	'Minho read a book.'		

2. Two Versions of Lambda Abstraction

The use of the λ-notation of type logic has been useful in the formal semantic analyses since Alonzo Church invented it in the 1940s. Especially, the rule of lambda abstraction has been argued to have two major functions. Lambda abstraction is crucial because it decomposes the known meaning of a complex structure into the denotations of its parts. That is, lambda abstraction sorts out the parts of the structure whose meaning is known, yielding the semantic contribution of the part whose meaning has previously been unknown.

Therefore, the rule of lambda abstraction is inevitable when it comes to compositionality. Also, lambda abstraction plays a significant role in the grammar in the respect that it turns saturated expressions into unsaturated expressions because the application of the lambda abstraction rule can create functions that demand an argument.

Heim and Kratzer (1998) have put forth what they call Predicate Abstraction defined in (3) along with two other major semantic rules: Function Application and Predicate Modification, defined in (4) and (5) respectively. According to them, the three rules basically account for all the ways the meanings of parts are combined to generate the meaning of the whole expression.

(3) Predicate Abstraction (Heim and Kratzer (1998: 107))
If α is a branching node and β and γ its daughters, when β is a relative pronoun or β = "such," then $[[\alpha]] = \lambda x \in D.[[\gamma]]^x$

(4) Function Application (Heim and Kratzer (1998: 49))
If α is a branching node, $\{\beta, \gamma\}$ is the set of α's daughters, and $[[\beta]]$ is a function whose domain contains $[[\gamma]]$, then $[[\alpha]] = [[\beta]]([[\gamma]])$.

(5) Predicate Modification (Heim and Kratzer (1998: 65))
If α is a branching node, $\{\beta, \gamma\}$ is the set of α's daughters, and $[[\beta]]$ and $[[\gamma]]$ are both in $D_{<e,t>}$, then, $[[\alpha]] = \lambda x \in D_e.\ [[\beta](x) = [[\gamma]](x) = 1$.

The two rules described in (4) and (5) are not highly controversial except the fact that the rule of Predicate Modification in (5) is applied in a more loosened form in many practices of formal semantics than Heim and Kratzer's original statement. Yet, the rule in (3) that is responsible for lambda abstraction has raised many questions due to its strict rigidity.

Bittner (1994) is one of the studies that provides an alternative form of lambda abstraction whose definition is closer to Montague's lambda abstraction rule. Bittner (1994) defines the rule of lambda abstraction as in (6). One of the fundamental reasons of putting forth the rule in (6) is to let it apply cross-linguistically.

(6) Bittner's λ-abstraction rule (variable binding rule)[1]
Let α have a translation [[α]] and let the index 'i' be the index of either α or a sister of α, and let [[α]] contain a variable u with index 'i.' Then, λu_i.[[α]] is a translation of α.

According to the rule in (6), a variable that is born with an index can be lambda-abstracted either when the expression that contains the variable has the index or when the syntactic sister of the expression carries the same index. Yet, since the rule is defined in a disjunctive way, under the general assumption that indices are allowed to be percolated along the syntactic trees, the rule in (6) permits any variable with an index to freely be λ-abstracted. That is, the rule in (6) does not restrict the occurrence of lambda abstraction only to the condition that is strictly co-indexed via a movement.

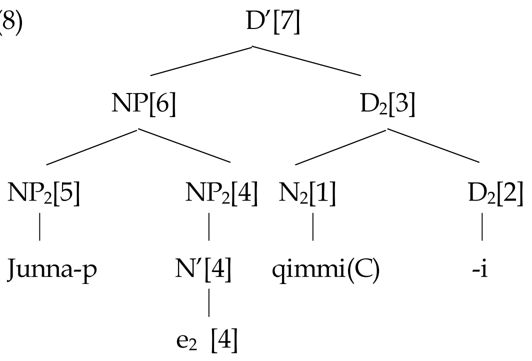
Let me clarify what the rule in (6) can account for with examples. Bittner (1994) explains the possessed nominal construction in West Greenlandic Inuit, using her rule in (6). This is one of the constructions that Bittner's lambda abstraction rule defined in (6) can easily account for. The syntactic structure of the possessed nominal phrase in (7) is illustrated in (8).[2]

(7)	Juuna-p	qimmi-i
	J-ERG	dog-3sgERG.pl
	'Juuna's dogs'	

1) Bittner's (1994) original statement of the rule is as follows: Let α have a translation <ε, σ>, let i be the index of either α or a sister of α, and let $u_i \in \sigma$. Then <λu_i[ε], σ-{u_i}> is a translation of α. The definition given in (6) is the Bittner's rule rephrased by Zimmermann (2002). For the sake of clarity, I use the rephrased definition in this paper.

2) Bittner (1994) assumes noun incorporation for the possessed construction in WG Inuit. However, this paper will not discuss the specific syntactic treatment in detail since it is not the main issue of this paper.

(8)



In the semantic derivation[3)] for (7) that is illustrated in (9), Bittner's (1994) lambda abstraction rule in (6) applies in node [6]. Only after this remedy takes place, the complex denotation of the possessed nominal phrase can successfully be derived from the meanings of its parts in a principled manner.

(9) Semantic Derivation for (7)

$[[1]] = {*}dog$

$[[2]] = \lambda Q \lambda x[\sigma y[of(Q)(x)(y) \ \& \ \neg At(y)]]$

$[[3]] = \lambda x[\sigma y[of({*}dog)(x)(y) \ \& \ \neg At(y)]]$

$[[4]] = O_2$

$[[5]] = j$

$[[6]] = \lambda O_2[O_2(j)]$

$[[7]] = [\sigma y[of({*}dog)(j)(y) \ \& \ \neg At(y)]$

Furthermore, Zimmermann (2002) provides an example that Bittner's rule in (6) can account for but Heim and Kratzer's rule in (3) cannot easily apply. Heim and Kratzer's rule of lambda abstraction relies on the movement of some element. Contrary to it, Bittner's rule in (6) is insensitive to whether or not a movement has occurred. The indifference to movements makes the rule in (6) a more liberated version of the rule in (3). The Dutch hanging topic construction in (10) shows us the case that satisfies the conditions of Bittner's rule in (6) but that does not involve movements.

3) In the semantic derivation illustrated in (9), the * operator is a sum operator while *At* stands for atoms.

(10) Jan$_i$, die$_i$ heb ik gisteren ontmoet.
Jan that-one have I yesterday met
'(Talking about) Jan, I have met him yesterday.'

The hanging topic construction in (10) includes a topic element that is placed in the left territory of the main clause and this constituent is picked up by a pronoun in the pre-verbal position. Zimmermann (2002) claims that the hanging topic *Jan* is, typically, base-generated as a left-peripheral 'satellite' and, to factor in the denotation of *Jan* in (10), a rule like Bittner's lambda abstraction is necessary.

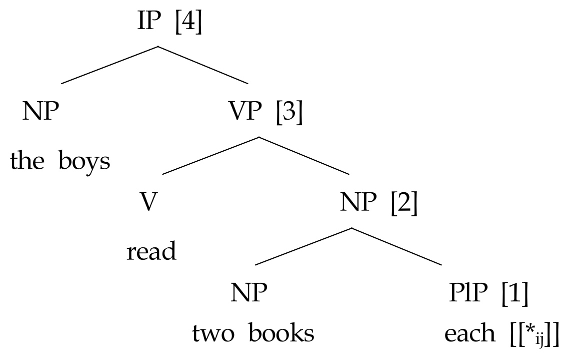
More specifically, the semantic derivation of (10) can be illustrated as in (11). The denotation of the CP is provided in (11a). Since there is a variable carrying an index in the denotation of the CP, the structure meets the licensing condition for the rule in (6). The rule of lambda abstraction yields a function as in (11b) and it applies to the denotation of *Jan* and derives the right interpretation for (10) as in (11c). Yet, since Heim and Kratzer's rule must assume a movement, it cannot properly analyze the example in (10).

(11.a) [[that one$_i$ have I yesterday t_i met]] = $\exists\lambda e[met(I, y_i, e)]$
(11.b) [[that one$_i$ have I yesterday t_i met]] =$\lambda y_i.\exists e[met(I, y_i, e)]$
(11.c) [[Jan$_i$ that one$_i$ have I yesterday t_i met]] =$\exists e[met(I, jan, e)]$

Next, I would like to illustrate that Bittner's lambda abstraction rule has a broader range of application than Heim and Kratzer's predicate abstraction rule with the example that I have dealt with in my previous work, Joh (2008). This example will show us that Bittner's rule is useful in compositionally combining elements that are in distance. For the semantic derivation of the sentence in (12a), the distributive particle *each* translates into a pluralization operator that has three variables. The denotation of *each* needs to be λ-abstracted before the free variables can be filled with the right value. As shown in (12b), two occurrences of lambda abstraction must turn saturated expressions into unsaturated expressions and the rule at play is in the spirit of Bittner (1994).

(12.a) The boys read two books each.

(12.b)



$$[[1]] = \lambda P.\forall z[z \in Z_i \rightarrow \exists x[P(x)\ \&\ R_j\ (x)(z)]]$$
$$[[2]] = \forall z[z \in Z_i \rightarrow \exists x[\uparrow(\text{two books})\ (x)\ \&\ R_j\ (x)(z)]]$$
$$[[2]] = \lambda R_j.\forall z[z \in Z_i \rightarrow \exists x[\uparrow(\text{two books})\ (x)\ \&\ R_j\ (x)(z)]]$$
$$[[3]] = \forall z[z \in Zi \rightarrow \exists x[\uparrow(\text{two books})\ (x)\ \&\ \text{read}\ (z, x)]]$$
$$[[3]] = \lambda Z_i.\forall z[z \in Z_i \rightarrow \exists x[\uparrow(\text{two books})\ (x)\ \&\ \text{read}\ (z, x)]]$$
$$[[4]] = \forall z[z \in [[\text{the boys}]] \rightarrow \exists x[\uparrow(\text{two books})\ (x)\ \&\ \text{read}\ (z, x)]]$$

To be more specific, in the derivation illustrated above, the denotation of the pluralization operator first applies to its syntactic sister. Then, the variable *R* with the index *j* gets λ-abstracted and the value of the *R* is determined by the main verb *read*. The distributive antecedent *the boys* is factored in via lambda abstraction over the index *i* and function application. The semantic derivation illustrated in (12b) precisely derives the reading of (12a) 'each of the boys read a group of two books,' deriving the ∀∃-structure compositionally.4)

4) Briefly speaking, I derive the distributive reading of the sentence in (12a) as in (12b) because distributivity is reduced to plurality. Distributivity involved in (12a) has traditionally been treated simply by applying Quantifier Raising or a scope mechanism. Yet, these treatments miss an important fact that distributivity is plurality that Landman (2000) introduces to the grammar. Thus, I have incorporated the ∀∃-structure in the denotation of the pluralization operator that is evoked by the distributive quantifier *each*. You can refer to my previous work for further details.

3. Scrambling and Its Formal Semantic Treatment

In the previous section, we have seen that Bittner's rule can give rise to a semantic function when an indexed free variable is embedded in a given formula and thus helps to derive the interpretation of the whole expression based on the meanings of its parts in a compositional way. We have also observed that Bittner's lambda abstraction rule can occur when there is no possible movement and hence explain a construction that would otherwise remain unexplained. The circumstance in which discrete entities have a dependent relationship to one another is another case that Bittner's rule can easily deal with but Heim and Kratzer's rule cannot explain adequately.

Overall, lambda abstraction proposed by Bittner (1994) can be used to ensure the interpretability of otherwise uninterpretable expressions as a cross-linguistic rule since, in terms of her rule, saturated expressions can be opened up rather freely. In a word, the application of Bittner's rule is a lot more liberated than that of Heim and Kratzer's (1998) rule which depends on indices that are introduced only by some sort of movement. Yet, the liberated nature of Bittner's rule should not concern us too much since the degree of freedom that we see in the rule is attested by the freedom we observe in real linguistic data.

In addition to the three examples discussed in the previous section, there is a compelling linguistic fact that shows us that Bittner's version of the lambda abstraction rule is truly at play in the grammatical system. At syntax, we can easily observe that arguments can freely be arranged. This phenomenon is generally called scrambling. In brief, I claim that the free arrangements of syntactic objects are essentially motivated by the semantic rule that freely creates a semantic function on the basis of an indexed free variable present in a given formula. In this section, I will primarily discuss a significant linguistic phenomenon that can lead us to the conclusion that lambda abstraction that Heim and Kratzer (1998) propose must be rendered into a form that is less restrictive just as Bittner (1994) argues.

Many languages such as Korean, Japanese, Hindi, Turkish, and German allow syntactic arguments to be scrambled in a highly flexible manner. In these free word order languages, constituents can appear in a wide variety of surface orders without making a significant change to the core interpretation of the sentence. What I primarily argue is that the high flexibility of syntactic constituents manifested by scrambling in a number of languages is fundamentally attributed to the corresponding flexibility in semantics, the degree of flexibility incorporated in the lambda abstraction rule advanced by Bittner (1994).

As shown below, the linear ordering of each argument of the verb *cwuta* 'give' can be shifted, with the underlying semantics maintained and the grammaticality of the sentence unaffected. The permutation of the three arguments of the verb *cwuta* 'give' provides us with six different sentence types as listed from (13a) to (13f). Evidently, the sentences in (13) deliver the same core semantics, regardless of the different surface orders of each syntactic constituent.

(13.a) Minho-ka Yuna-eykey sathang-ul cwu-ess-ta. [S > I. > D]
Minho-Nom Yuna-Dat candy-Acc give-Pst-Dec
'Minho gave Yuna candy.'
(13.b) Minho-ka sathang-ul Yuna-eykey cwu-ess-ta. [S > D > I]
Minho-Nom candy-Acc Yuna-Dat give-Pst-Dec
'Minho gave Yuna candy.'
(13.c) Yuna-eykey Minho-ka sathang-ul cwu-ess-ta. [I > S > D]
Yuna-Dat Minho-Nom candy-Acc give-Pst-Dec
'Minho gave Yuna candy.'
(13.d) Yuna-eykey sathang-ul Minho-ka cwu-ess-ta. [I > D > S]
Yuna-Dat candy-Acc Minho-Nom give-Pst-Dec
'Minho gave Yuna candy.'
(13.e) Sathang-ul Yuna-eykey Minho-ka cwu-ess-ta. [D > I > S]
Cany-Acc Yuna-Dat Minho-Nom give-Pst-Dec
'Minho gave Yuna candy.'
(13.f) Sathang-ul Minho-ka Yuna-eykey cwu-ess-ta. [D > S > I]
Candy-Acc Minho-Nom Yuna-Dat give-Pst-Dec
'Minho gave Yuna candy.'

With the lambda abstraction rule proposed by Heim and Kratzer (1998), for sentences listed in (13), we need to have as many lexical denotations as the scrambled structures that one and the same verb can have. That is, the same verb *cwuta* 'give' that appears in the sentences in (13) is to have six different denotations that reflect the different word orders, as illustrated from (14a) to (14f), as long as we do not make the syntax complicated against one of the central tenets of the Minimalist view.

(14.a) $\lambda Y \lambda Z \lambda X. \exists e \in gave: Ag(e) \in X \;\&\; Th(e) \in Y \;\&\; Re(e) \in Z$
(14.b) $\lambda Z \lambda Y \lambda X. \exists e \in gave: Ag(e) \in X \;\&\; Th(e) \in Y \;\&\; Re(e) \in Z$
(14.c) $\lambda Y \lambda X \lambda Z. \exists e \in gave: Ag(e) \in X \;\&\; Th(e) \in Y \;\&\; Re(e) \in Z$
(14.d) $\lambda X \lambda Y \lambda Z. \exists e \in gave: Ag(e) \in X \;\&\; Th(e) \in Y \;\&\; Re(e) \in Z$
(14.e) $\lambda X \lambda Z \lambda Y. \exists e \in gave: Ag(e) \in X \;\&\; Th(e) \in Y \;\&\; Re(e) \in Z$
(14.f) $\lambda Z \lambda X \lambda Y. \exists e \in gave: Ag(e) \in X \;\&\; Th(e) \in Y \;\&\; Re(e) \in Z$

However, with the liberated lambda abstraction rule put forth by Bittner (1994), we can simplify the lexicon with just one denotation described in (15), instead of imposing six different denotations on one and the same verb. That is, relying on Bittner's rule in (6), we can reduce the six distinct denotations listed in (14) into just one denotation in (15)[5] for the verb *cwuta* 'give.' If we think about the number of verbs in natural languages and the number of all the surface word orders that they can allow, the simplifying effect that Bittner's rule can bring to us is indeed enormous.

(15) $\exists e \in gave: Ag(e) \in X_i \;\&\; Th(e) \in Y_j \;\&\; Re(e) \in Z_k$

Let me explain more specifically how efficiently Bittner's rule can address various word orders that one verb allows with examples. The denotation in (15) has the information that the verb *cwuta* 'give' has three arguments but the exact

5) According to the denotation given in (15), the verb denotes an under-specified proposition and this seems right since the meaning of a verb anticipates what kinds of arguments it will take even though the information is not fully specified in the lexicon. Another point to make regarding the denotation in (15) is that the so-called canonical word order in scrambling languages is not the reflection of the fixed structural order but merely the reflection of the semantic structure represented in the lexicon. As shown in (15), thematic roles are placed in a certain order. The canonical word order in scrambling languages simply mirrors the order of these thematic roles.

syntactic order of the arguments is under-specified. The order of each argument is to be determined at the syntax when the verb meets the right constituent in the process of semantic composition. As will shortly be illustrated, the variable with an index inside the denotation of the verb *cwuta* 'give' is a potential target of lambda abstraction and when it meets the proper constituent in the due derivational process, it gets λ-abstracted. In other words, the under-specified order of each argument of the verb *cwuta* 'give' is determined only when the appropriate syntactic information is provided and, along with the flexible syntax, the flexible semantic composition can go through since lambda abstraction in (6) permits the variables whose order are not pre-determined to be λ-abstracted at the right moment.

For instance, the sentences in (13a) and (13d) can semantically be processed as in (16a) and (16b), respectively. As you can see, just one denotation of the verb *cwuta* 'give' can explain various syntactic arrangements of the arguments it takes, thanks to the flexible rule of lambda abstraction. This is highly desirable considering that all the sentences in (13) share the same underlying interpretation although it might be true that some different discourse factors may kick in.

(16.a) [[cwu-ess-ta]]
$= \exists e \in \text{gave}: Ag(e) \in X_i \ \& \ Th(e) \in Y_j \ \& \ Re(e) \in Z_k$
$\Rightarrow \lambda Y_j. \exists e \in \text{gave}: Ag(e) \in X_i \ \& \ Th(e) \in Y_j \ \& \ Re(e) \in Z_k$
<Lambda Abstraction>
[[sathang-ul cwu-ess-ta]]
$= \exists e \in \text{gave}: Ag(e) \in X_i \ \& \ Th(e) \in \text{CANDY}$
$\& \ Re(e) \in Z_k$
<Function Application>
$\Rightarrow \lambda Z_k. \exists e \in \text{gave}: Ag(e) \in X_i \ \& \ Th(e) \in \text{CANDY}$
$\& \ Re(e) \in Z_k$
<Lambda Abstraction>
[[Yuna-eykey sathang-ul cwu-ess-ta]]
$= \exists e \in \text{gave}: Ag(e) \in X_i \ \& \ Th(e) \in \text{CANDY}$
$\& \ Re(e) \in \text{Yuna}$
<Function Application>

$\Rightarrow \lambda X_i. \exists e \in$ gave: Ag(e) $\in X_i$ & Th(e) $\in$ CANDY
& Re(e) $\in$ Yuna
<Lamba Abstraction>
[[Minho-ka Yuna-eykey sathang-ul cwu-ess-ta]]
= $\exists e \in$ gave: Ag(e) $\in$ Minho & Th(e) $\in$ CANDY
& Re(e) $\in$ Yuna
<Function Application>

(16.b) [[cwu-ess-ta]]
= $\exists e \in$ gave: Ag(e) $\in X_i$ & Th(e) $\in Y_j$ & Re(e) $\in Z_k$
$\Rightarrow \lambda X_i. \exists e \in$ gave: Ag(e) $\in X_i$ & Th(e) $\in Y_j$ & Re(e) $\in Z_k$
<Lambda Abstraction>
[[Minho-ka cwu-ess-ta]]
= $\exists e \in$ gave: Ag(e) $\in$ MINHO & Th(e) $\in Y_j$ & Re(e) $\in Z_k$
<Function Application>
$\Rightarrow \lambda Y_j. \exists e \in$ gave: Ag(e) $\in$ MINHO & Th(e) $\in Y_j$
& Re(e) $\in$ Zk
<Lambda Abstraction>
[[sathang-ul Minho-ka cwu-ess-ta]]
= $\exists e \in$ gave: Ag(e) $\in$ MINHO & Th(e) $\in$ CANDY
& Re(e) $\in Z_k$
<Function Application>
$\Rightarrow \lambda Z_k. \exists e \in$ gave: Ag(e) $\in$ MINHO & Th(e) $\in$ CANDY
& Re(e) $\in Z_k$
<Lambda Abstraction>
[[Yuna-eykey sathang-ul Minho-ka cwu-ess-ta]]
= $\exists e \in$ gave: Ag(e) $\in$ MINHO & Th(e) $\in$ CANDY
& Re(e) $\in$ YUNA
<Function Application>

With regard to the formal semantic analysis of the phenomenon of scrambling provided above, I would like to point out three advantages that Bittner's lambda abstraction rule can give us, overcoming some limitations that Heim and Kratzer's rule faces. First of all, the approach introduced above employing Bittner's lambda abstraction rule reduces the avoidable complexity in the lexicon in two aspects. First, it helps us to cut down the number of separate lexical entries enormously, as mentioned before.

Secondly, it suggests us that the information as to the order of the arguments does not need to be specified in the denotation of each verb. At least in scrambling languages, arguments of a verb can basically be arranged in any order. Then, why do we restrict them in a specific order just to follow non-scrambling languages and make the lexicon complicated? If it is apparently and obviously the case that word order is not pre-fixed, then the most natural approach is that we don't pre-determine the word order in the lexicon unnecessarily.

This kind of theoretical move that tries to minimize the role of the lexicon has already been taken with regard to grammatical functions. Farrell (1998) claims that a specific grammatical function is determined only when contextually imposed profiling scenarios are provided. That is, only when a word appears in a verb position, it is realized as a verb. Likewise, when a certain lexical item appears in a position that the phrasal structure fixes it to be a noun, it becomes a noun. In other words, such lexical items as *dance, nurse, accordion,* etc. which can be either a noun or a verb are represented as the X category underlyingly. For example, an English word like *bag* which fluctuates between the noun and the verb has a semantic manifestation that is under-specified for the distinction between a noun and a verb in the lexicon. It later becomes either a noun or a verb at the moment when the sufficient syntactic information is provided. This approach seems to be on the right track since the distinction between nouns and verbs is arbitrary in nature without considering their configurational properties, as generally acknowledged. As a matter of fact, this idea originates in Chomsky (1970) who argues that the reliable notion of functional categories must be defined at a certain syntactic level.

Furthermore, the entire framework of a linguistic project generally known as Distributed Morphology that is represented by works of Halle and Marantz (1993, 1994) advocates this move in that they claim that the load of the lexicon must be minimized. In terms of this widely accepted view as well, the free syntactic alternations of arguments suggest us that underlyingly in the lexicon the information on the strict word order is under-specified, at least in scrambling languages. What it means is that the order of syntactic constituents are products of the syntax, not the lexicon, and Bittner's rule can make it possible for the idea to be implemented in the grammatical system since the rule allows us to formally analyze how the under-specified information concerning the order of the arguments can be specified at the sentential level.

In fact, Heim and Kratzer's (1998) rule can also simplify the lexicon when a number of movements are assumed. Yet, this is done only at the cost of making the syntactic level complex. That is, Heim and Kratzer's definition of lambda (predicate) abstraction can derive the meanings of various surface orders of arguments by postulating one denotation for one verb, as far as scrambling is concerned. However, instead of keeping the lexicon simple, they must complicate the syntax. When a noun phrase moves, it is canonically assumed that it leaves a trace behind. Thus, the arguments in their fixed order as shown in (14a) first consume the traces the movements have left behind one by one and later the moved elements can be factored in via the variables that are λ -abstracted. However, only by assuming free movements, Heim and Kratzer's (1998) rule can derive the interpretation of the scrambled sentence from the denotation of the verb that is defined for the unscrambled sentence.

Yet, many recent studies have correctly pointed out that scrambling should not be approached with movements. For instance, developing the idea of Boskovic and Takahashi (1998), Cho and Kim (2010) have argued that scrambled phrases are base-generated in their surface positions since the Last Resort principle under the minimalist framework bans any movement from taking place, unless it is strictly driven by obligatory morphological necessity. They further claim that, especially for clause-internal scrambling, neither syntactic nor LF movement should be applied.

In brief, Heim and Kratzer's rule faces a serious challenge in coping with scrambling if they want to make the lexicon simple. Yet, without making the syntax complex by resorting to movements that are against the fundamental principle of syntax, Heim and Kratzer's (1998) rule must make the lexicon complicated. In a word, whether it is the syntax or the lexicon, Heim and Krater's rule must keep a high degree of complications in the grammar. However, Bittner's rule has an advantage of avoiding this burdensome theoretical hassle altogether and is free of any problems that might be caused by certain syntactic assumptions.

Moreover, without having to assume any kinds of movements for scrambling, Bittner's rule can view scrambling not as a degenerate form that is deviated from the standard structure but as a natural linguistic phenomenon that is deeply rooted in a crucial rule that is independently justified. In a word, the form of lambda abstraction defined in Bittner's way fundamentally accounts for what induces scrambling in the first place. Bittner's lambda abstraction rule as one of the major semantic rules allows any free variable with an index to be λ-abstracted. This is basically how and why an argument can freely appear in any place in a sentence. That is, Bittner's lambda abstraction serves as the essential driving force that generates scrambling. Therefore, scrambling can be referred to as a phenomenon that is, in essence, semantically motivated. When the semantic rule of lambda abstraction in its relatively free style is realized syntactically, scrambling emerges. In a nutshell, Bittner's λ-abstraction rule can account for what motivates scrambling without complicating the lexicon or assuming any syntactic movements. Thus, in this approach, scrambling can be viewed as naturally as it gets.

Basically, I claim that Bittner's semantic rule explains a syntactic phenomenon. Looking at the opposite side of the same coin, we can also say that the occurrence of scrambling at syntax leads us to conclude that such a rule advanced by Bittner (1994) exists at semantics. Since syntax and semantics mutually feed each other, either interpretation is possible and legitimate. Regardless of which direction we take, Bittner's rule helps us to directly address the correlation between the flexible syntax and the flexible semantics.

A historical fact of the English language has some implications on this issue. In old English, scrambling was freely allowed. Yet, as time went by, English started to lose many features especially regarding case. When these features have begun to get expressed optionally or at least covertly, the English language could not allow its syntactic arguments to be freely arranged any more. What this seems to show us is that scrambling is something that can be permitted when the grammatical system is more sophisticated, in a sense. Given that, it is very odd to approach scrambling as something that is deviated from the canonical form that fixes the word order underlyingly since it might imply that the English language has changed from the derived structure to the underlying structure, contrary to our general intuition.

In fact, the change of the English language shows us that the direction is the opposite. English started as a scrambling language and came to be a non-scrambling language. Although other interpretations are possible, what this seems to suggest us is that scrambling languages should not be approached by placing additional principles or restrictions on the mechanisms which are ultimately designed for non-scrambling languages. Rather, scrambling must be treated as a phenomenon that stands independently and, as shown above, Bittner's rule is the tool that makes us achieve it in a very economical way.

Related to the point that scrambling has to do with the highly sophisticated system of case, I would like to discuss one more aspect of Bittner's rule. The grammatical device that makes it possible for various arguments to be scrambled is basically the case markers. Without them, much confusion can arise and scrambling cannot occur in such a free form.6) Since case markers overtly indicate which arguments are assigned to which syntactic or semantic roles, various constituents of a sentence can be ordered rather freely.

Lee (1993) has claimed that scrambling is a case-driven phenomenon. According to him, arguments can be arranged in any order only when the licensing conditions on case are met. Lee (2007) also argues that the free arrangements of syntactic constituents is attributed to the rich system of overt case markers. Otherwise, scrambling is impossible. As they precisely argue, case is one of the crucial features that enables to trigger scrambling. The current approach based on Bittner's rule can capture this property of scrambling as well.

Bittner's rule intrinsically utilizes indices, in contrast to Heim and Kratzer's rule that does not incorporate them unless a movement is involved. By having the inherent index system within the rule, Bittner's rule can essentially embrace the idea that scrambling is case-driven. Bittner's rule states that a free variable

6) Kiaer and Kempson (2005) discuss some restrictions on scrambling that must be explained by additional syntactic principles. Yet, I would like to point out that scrambling, if it occurs, basically arranges various arguments in a highly free way. The presence of some restrictions on scrambling are certainly what we must explain but it does not change the fact that the phenomenon of scrambling essentially shows us the powerful flexibility of syntax. Bittner's rule basically explains the underlying flexibility and additional restrictions on the syntactic flexibility must further be addressed by placing additional semantic principles outside of Bittner's rule.

can be λ-abstracted, when it carries an index. This index is the marker that helps to seek for a phrase that is appropriate for the value of the index-carrier. The variables can be filled with the right value since the indices ensure the variable to be λ-abstracted when the variable meets the proper constituent. In a word, one of the basic functions of indices is to search for the suitable match.

The indices that are placed in the denotation of a verb have the information concerning what thematic roles that the variables that they attach to play. Based on this, they are able to find which syntactic argument that they must be connected with at syntax. That is, the information regarding the thematic role that each variable plays is "compositionally visible" through indices. In the extension described in (15), the variable X plays the agent role while the variable Y plays the theme role. The variable Z is the Recipient, as indicated. What makes this information visible in the semantic process is the indices since their function is to find the right value that can be co-indexed with them.

Let's look at how indices work with pronominals. The pronoun *she* in the second sentence in (17) has an index. The index gathers all the information on the noun phrase that bears it in order to search for the right antecedent. It uses the gender feature as well as the number feature of the noun phrase it attaches to in order to find *Mary* as the suitable match that can be co-indexed with it.[7)]

(17) John and Mary went to the store. She*j* bought a book.

Just like this, the index in general helps the index-carrier to find the right value. In the case of the verbs, an index suffixed to a variable uses all the information regrading the variable such as thematic roles in order to find the right constituent it can be co-indexed with and what makes the co-indexation possible is the overt case marker attached to a syntactic constituent. That is, using the information on what thematic role a variable plays, the index finds the right syntactic constituent that can be associated with the thematic role of the variable. Generally, the association can systematically be established by a linking principle like (18) along with the hierarchies presented in (19).

7) I am basically following the dynamic view on indices. Yet, there is also a view that treats indices merely as syntactic objects that serve as the input to the interpretational process. Yet, the problem of this kind of view has been discussed in Nouwen (2003).

(18) Linking Principle (Jackendoff, 1990, p. 247)
Map the ordered theta-roles from the Lexical Conceptual Structure into the syntactic hierarchy from left to right.

(19.a) Thematic hierarchy (Jackendoff, 1990, p. 247)
Agent > Theme (> Recipient)

(19.b) Syntactic hierarchy (Jackendoff, 1990, p. 247)
Subject > Object (> Indirect Object)

Since the right value of each variable in the denotation of a verb is found in such a way, overt marking of case is important in cases of scrambling. Only when there is an overt case marker for each syntactic constituent, the indexed free variables in the denotation of the verb can find their right value on the basis of the linking principle. In this fashion, the index system in Bittner's rule helps us to address how scrambling works in the intricately interconnected grammatical system.

That is, scrambling is case-driven and the indices in Bittner's rule can capture this fact in an indirect way. Under the assumption that each argument of a verb is fixed with its thematic role while its word order being not pre-determined in scrambling languages, finding a right value for each variable becomes a task that can be done only with overt case markers. With Bittner's rule employed, each of the variables in the denotation of the verb carries an index which can help us to see which semantic role it plays. Only with this index system, the free variable in the denotation of a verb can activate its search engine with proper information and the search can be successful only when the syntactic constituents are overtly marked with case.

What this system predicts is that scrambling cannot be licensed when the syntactic constituents are not clear with their grammatical roles. This is evidently borne out. As claimed widely, scrambling occurs based on the case system. Thus, Bittner's rule not only explains why we have scrambling but also addresses the issue why some languages cannot allow it. When noun phrases do not carry overt case markers, the variables with an index in the under-specified denotation of the verb cannot find their right values so that scrambling cannot occur.

Scrambling languages commonly stand out by their rich system of case compared to non-scrambling languages that characteristically do not have the highly developed system of case. Heim and Kratzer's rule has no way to captivate this fact but Bittner's rule can express it in a principled manner. That is, Heim and Kratzer's rule cannot explain why scrambling is sensitive to the presence or the absence of the overt case markers in general. If the lexicon is complex, the right values of the variables are pre-determined by the strict order of the variables. If the syntax is complex, the values of the variables are strictly fixed by the co-indexation that is introduced by movements. Thus, in Heim and Kratzer's system, it is hard to find a place to express the case factor for scrambling. Yet, Bittner's rule can explain why overt case marking essentially licenses the free form of scrambling. With the lexicon that is under-specified with respect to the syntactic order of the arguments, the search for the right values for the free variables can successfully be done only when the syntactic constituents are overtly marked with their grammatical functions since indices having the information on thematic roles can find the proper values only at the presence of overt case markers through the general linking rule.

Having the index system in a semantic rule seems an unnecessary complication at first glance, so it looks like Heim and Kratzer's rule without the inherent index system is more economical, at least in this respect. Yet, I would like to emphasize that the index system included in Bittner's rule is the necessary complexity and it places the inevitable necessity in the right place. By embedding the system of indices, Bittner's rule is placed in a better position to approach scrambling. It explains why case markers matter when it comes to scrambling.

Lastly, I would like to make two general points regarding my current work. First, there are many researchers who are overly concerned about loosening compositionality by employing various devices such as flexible semantic rules and covert operations, etc. Of course, the concerns should always be kept in mind but the real problem we are facing is that, without those progressive and innovative mechanisms, the principle of compositionality must be dropped altogether since there are so many empirical data that do not simply work with the rigid definition of compositionality. Thus, the question is whether we are going to give up compositionality or keep it by liberating it a little bit. In this paper, I have shown how we can choose the second option. In other words, my current work that supports a loosened form of a semantic rule is one of the endeavors that has been put into in order to keep compositionality, far from ruining it in any sense.

Second, I would like to point out that my current work is also important in that it reveals what kind of relationship there is between syntax and semantics in general. Also, many researchers have argued that semantics is merely read off from syntax since syntactic structures precede the semantic process. Based on this, they have claimed that semantics must entirely be dependent on syntactic structures. However, my current work challenges this assumption, showing that semantics can motivate the syntactic structure in a non-trivial way. Thus, one of the implications of my work is that semantics can also feed syntax just like syntax can feed the semantic process.

4. Conclusion

In this paper, I have primarily weighed two semantic proposals regarding lambda abstraction and came to the conclusion that Bittner's (1994) rule is much more desirable than Heim and Kratzer's (1998) definition of lambda abstraction. The supporting evidence has been found in the important linguistic phenomenon of scrambling which is used as a cover term for the grammatical process that evokes various patterns of word order, without shifting the core sense of the sentence. With Heim and Kratzer's rule of lambda abstraction, we need to have as many separate lexical entries as the different orders of syntactic arguments, unless the syntax is made complicated against one of the basic tenets of syntax. This unnecessarily complicates the grammatical system in an enormous way since it means that we need to multiply the number of a verb whenever it occurs with different argument structures. However, employing Bittner's rule, we can simply avoid this problem by assigning one denotation to one verb as long as the scrambling variations are concerned.

The immediate benefit of employing Bittner's (1994) rule is that the grammar gets simpler to a high extent, in line with general assumptions on the simple lexicon. In addition to this, there are two more merits. Bittner's rule helps us to approach scrambling not as a phenomenon which is deviated from the standard form but as one of the syntactic facts that is semantically motivated. In fact, the fundamental driving force of scrambling is one of the productive rules that we have at the semantic level. That is, the semantic fact that variables with an index can freely be λ-abstracted essentially explains the syntactic fact that arguments of a verb can freely be arranged. Furthermore, the approach based on Binnter's rule that has the index system as an essential part of it correctly reflects the aspect that scrambling is basically possible at the presence of case markers. In sum, Bittner's rule explains not only why scrambling occurs but also when it can and cannot occur.

References

Bittner, M. (1994). Cross-linguistic Semantics. *Linguistics and Philosophy,* 17, 53-108.

Boskovic, Z. and D. Takahashi. (1998). Scrambling and Last Resort. *Linguistic Inquiry,* 29, 347-366.

Cho, J. H. and O. H. Kim (2010). Scrambling Without Syntactic Movement. *Studies in Generative Grammar,* 10, 151-179.

Chomsky, N. (1970). Remarks on Nominalization. In R. Jacobs and P. Resenbaum (Eds.), *Readings in English Transformational Grammar* (pp. 184-221). Mass: Waltham.

Farrell, P. (1998). Noun/Verb Functional Shift: A Cognitive Analysis. In T. Haukioja (Ed.), *Papers from the 16th Scandinavian Conference of Linguistics* (pp. 101-112). Department of Finnish and General Linguistics, University of Turku.

Halle, M. and A. Marantz. (1993). Distributed Morphology and the Pieces of Inflection. In K. Hale and S. Keyser (Eds.), *The View From Building 20* (pp. 111-176). Cambridge: MIT Press.

Halle, M. and A. Marantz. (1994). Some Key Features of Distributed Morphology. In *MITWPL 21: Papers on Phonology and Morphology* (pp. 275-288). Cambridge: MITWPL.

Heim, I. and A. Kratzer. (1998). *Semantics in Generative Grammar.* Oxford: Blackwell.

Jackendoff, R. (1990). *Semantic Structures*. Cambridge: The MIT Press.

Joh, Y. K. (2008). *Plurality and Distributivity.* Unpublished doctoral dissertation. University of Pennsylvania.

Kiaer, J. and R. Kempson. (2005). Pro-active Parsing of Korean Scrambling. In *Proceedings of the 24th West Coast Conference on Formal Linguistics* (pp. 209-217). Somerville, MA: Cascadilla Proceedings Project.

Landman, F. (2000). *Plurality and Events: The Jerusalem Lectures.* Dordrecht: Kluwer.

Lee, Y. S. (1993). *Scrambling as Case-driven Obligatory Movement*. Unpublished doctoral dissertation. University of Pennsylvania.

Lee, E. S. (2007) A Semantic Restriction on Scrambling in Korean. In *LSO Working Papers in Linguistics 7: Proceedings of CUIGL* (pp. 109-123).

Nouwen, R. (2003). *Plural Pronominal Anaphora in Context: Dynamic Aspects of Quantification.* Utrecht: LOT.

Zimmermann, M. (2002). *Boys Buying Two Sausages Each*. Unpublished doctoral dissertation. University of Amsterdam.

Yoon-kyoung Joh
#204-704 Samsung Raemian APT Imun-1-dong
Dongdaemun-gu Seoul, 130-080, South Korea
E-mail: ykjoh@hotmail.com
Phone: 02-965-2221

Received: 18 January, 2011
Revised: 08 February, 2011
Accepted: 14 March, 2011