Anaphor-Antecedent Asymmetry: A Conceptual Necessity?

Edward L. Keenan
UCLA

> "... A still further step would be to show that the basic principles of language are formulated in terms of notions drawn from the domain of (virtual) conceptual necessity" (p.6)
>
> N. Chomsky, 1992

We consider a condition **F**, which relates the syntactic structures[1] of a natural language and their interpretation in *any* other system, in particular in ones, considered here, we may plausibly call semantic or "conceptual-intentional".

F is general: it is neither language nor structure specific, nor even rule, structure, or representation *type* specific. Symbolic systems L which satisfy **F** are ones which are, in a sense made precise in 3, *faithful* to the syntactic structure of L. Accordingly we shall refer to **F** as a *Fidelity* Principle. Mnemonically:

(1) **F**

Semantic interpretation is faithful to syntax[2]

I claim that **F** as defined in 3 expresses a very general truth about natural language. In the terms of CH-92, **F** is the interpretative analogue of a largely language invariant "computational procedure" which derives the syntactic structures[1] of a language L from the partially L-specific lexical items and morphology. By contrast, symbolic systems that fail **F** lack a regular relation between form and interpretation. Such "systems" seem to me conceptually ill suited to the representation of information, and would be decidedly unlanguage-like. Thus I offer **F** as a linguistic candidate for a "(virtual) conceptual necessity" (without suggesting that **F** was intended in Chomksy 1992 (CH-92)).

To support **F** and to explicate the notions in terms of which it is formulated, we argue that it derives (2).

(2) **Anaphor-Antecedent Asymmetry (AAA)**

> In minimal instances of the *Anaphor-Antecedent* relation we cannot interchange an anaphor and its antecedent preserving both syntactic form and logical meaning

Utpal Lahiri and Adam Wyner (eds.), *SALT III*, 117-144, Ithaca, N.Y.: Cornell University.

1. Anaphor-Antecedent Asymmetry (AAA).

A-expressions are ones that must be interpreted as *referentially dependent* in a certain way (see 2). Lexical A-expressions will sometimes be called *anaphors*. Some examples of A-expressions are italicized in (3).

(3) a. Some student's teacher blamed *himself* for the accident.
b. One of the students criticized *everyone but himself*
c. No student criticized *both himself and his teacher*

Since A-expressions are semantically defined, the following query makes sense:

Query 1 (Q1) Given a grammar G for a language L,

a. Is the set of A-expressions of L *syntactically identifiable in terms of* G ?
b. If not, is it *lexically identifiable* (= computable given G and the set lexical A-expressions.)?

To vary terminology in contexts like **Q1a** we may replace *syntactically* with *structurally* and *identifiable* with *definable* or *characterizable*. A fully general definition of this notion is given in **2.2**. We also provide a semantic definition of *A-expression* and exhibit a formal language *Little English* in which the set of A-expressions is not syntactically definable in terms of the grammar we provide. But that set is lexically identifiable. The example shows that:

i. Semantically defined sets may fail to be syntactically definable. So **Q1a** is not circular or trivial. Natural languages could fail it; and

ii. The ability to structurally identify A-expressions is not a conceptual necessity. The simplicity and naturaleness of the example support this.

Much empirical work[3] (esp. the "skeptical" Keenan 1987, 1988) leads me to think that natural languages *do* structurally identify their A-expressions. So I think that **Q1a**, and a fortiori **Q1b**, are to be answered affirmatively (though in the absence of adequate grammars no firm conclusion can be drawn). But a full yea to **Q1** does not disturb the truth of (ii). (ii) only claims that the structural identifiability of A-expressions is not *conceptually* necessary.

By contrast, I claim that the set of logically true sentences of (real) English is *not* syntactically definable in terms of any adequate grammar for English. (We think of a sentence as being *logically true* iff it is true no matter what the world is like). Here is one argument (among infinitely many) that supports this claim

(and helps us understand it better): (4a,b) should be *syntactically isomorphic* (= have the same syntactic structure) on any adequate grammar of English.

(4) a. Either fewer than thirty or else more than ninety linguists were martyred

b. Either fewer than ninety or else more than thirty linguists were martyred

I know of no grammaticality based property which distinguishes (4a) from (4b). Such a property would have to distinguish *thirty* from *ninety*. So these Ss appear to be syntactically identical; they are built in the same way, differing only trivially with regard to lexical choice. But (4b) is logically true: every number, in particular the number of linguists who were martyred, is either less than 90 or greater than 30. But (4a) is not logically true. It is false in any situation in which the number of linguists martyred is exactly 37.

Thus just knowing the syntactic form of an English sentence is not sufficient to decide whether it is logically true or not. Ss like (4a,b) have the same form but differ wrt their logical truth This can happen because part of what determines the truth of a sentence is the meanings of its lexical items and we can often find semantically distinct lexical items which cannot be syntactically distinguished. Indeed, the notion *grammatical category* serves, in part, to class together expressions which can be discriminated semantically but not syntactically.

In *Little English*, *himself* is an A-expression but *him* is not. But the syntax of this language does not distinguish between *him* and *himself* -- a derivation beginning with one can always have it replaced by the other preserving well formedness (but not of course interpretation). So in this L we find syntactically isomorphic expressions (e.g. *himself* and *him*; *both himself and john* and *both him and john*, ...) which differ with regard to being A-expressions. That is, the set of A-expressions is not syntactically identifiable in *Little English*.

Real English seems to differ from our artificial example in this regard. *himself* can not always replace *him* initially in a derivation preserving grammaticality: *Both John and Bill criticized him* is grammatical but *Both John and Bill criticized himself* is not. I suspect that the only expression *himself* is isomorphic to in English is *himself* itself (but we need an adequate grammar to support this!). We shall later characterize "grammatical words" as those expressions that are isomorphic only to themselves.

It seems then that the property of being interpreted anaphorically is "coded" in the syntax of English in a way in which the property of being logically true is not, a result in accordance with the "naive" intuitions of generative grammarians.

Secondly, as per work in Binding Theory, we understand that the anaphor *himself* in (5) stands in the *is anteceded by* (= AA) relation to *each student's advisor*. It cannot stand in that relation to *each student* or to *noone* in (5).

(5) Noone thought that each student's advisor disqualified himself

Query 2 (Q2)

a. Does every natural language L have a grammar G in terms of which the AA relation in L is syntactically definable?
b. If yes, then does there exist a *uniform* definition -- one that applies in the same way in all Ls?

Q2a says that given an occurrence of an A-expression α in an expression σ we should be able to specify in terms of the grammar G just which (occurrences of) expressions in σ are possible antecedents for α. **Q2b** says that this computation is executed in the same way in all Ls.

Much work in generative grammar[3] is consistent with an affirmative answer to **Q2a**. Keenan (1987, 1988, 1991) plus a few remarks below suggest that **Q2b** should be answered negatively. But we do not pursue this question here.

Rather, we investigate the notion of *asymmetry* in **AAA**. Our intuition is that languages are not indifferent as to the *relative* presentation of A-expressions and their antecedents. They care which is which. But how to say this rigorously?

We begin with an example. An adequate grammar of English will enable us to say that (6a,b) have the *same* syntactic structure (though different grammars may differ as to just what that structure is):

(6) a. John criticized Bill b. Bill criticized John

(6a,b) differ just by choice of lexical item -- once those choices are given the derivation and interpretation of the two Ss proceeds identically. So showing that (6a,b) have the same structure will involve matching up the steps in their derivations, beginning with the matching of the lexical items: *John* with *Bill*, *criticized* with *criticized*, and *Bill* with *John*. The matched expressions are images of each other under the *structure preserving maps* (isomorphisms) which say that (6a,b) have the same structure (= are isomorphic). Now consider:

(7) a. John criticized himself b. Himself criticized John

Could we have a language like English except that in addition to Ss like (7a), we systematically find logical paraphrases like (7b) which are not only grammatical but are syntactically isomorphic to (7a), just as (6b) is to (6a)?

The **AAA** says no. But is this claim is consistent with linguistic reality?

As regards Modern American English, Ss like (7b) are simply ungrammatical and so will satisfy **AAA** in virtue of exhibiting some sort of structural difference from (7a). In some varieties of English however, notably Irish English[4], (7b) is perfectly grammatical. But it still satisfies **AAA** since it does not exhibit the binding relations of (7a) and so is not a logical paraphrase of it. Rather the subject *himself* in (7b) is deictically interpreted as something like "the most prominent male" in the context of utterance. (And in *Wait a minute, herself is getting herself ready* (J. McCloskey (pc); See Keenan 1988) the first occurrence of *herself* is deictically interpreted as the prominent female in context).

Thus our intuition of asymmetry could not have been captured simply by "cannot be interchanged preserving grammaticality".

Tougher nuts concern the freedom of anaphor-antecedent order ("scrambling")[5] in Ls like Bengali, Hindi, and Korean. Arguments of transitive verbs in these head-final Ls are distinguished by overt case marking, their relative order preverbally being rather free, even when one is an anaphor whose antecedent is properly quantified (9) or interrogative (10). (Nominative marking is *-ka* following a vowel, *-i* following a consonant).

(8) a. John-i caki-casin-ul pinanhayssta KOREAN
John-nom self- -acc criticized
John criticized himself.

b. caki-casin-ul John-i pinanhayssta
self- -acc John-nom criticized
John criticized himself

(9) a. Nwukwunka(-ka) caki-casin-ul pinanhayssta
someone(-nom) self- -acc criticized
Someone criticized himself

b. Caki-casin-ul nwukwunka(-ka) pinanhayssta
self- -acc someone(-nom) criticized
Someone criticized himself

(10) a. Nwuka caki-casin-ul pinanhayss-ni
Who self- -acc criticized-Q
Who criticized himself?

b. Caki-casin-ul nwuka pinanhayss-ni
self- -acc who criticized-Q
Who criticized himself?

So in Korean, A-expressions like *caki-casin-ul* may be interchanged with their antecedents preserving both grammaticality and logical meaning.

The **AAA** claims that the (a,b) pairs in (8) - (10) are not syntactically isomorphic. While in agreement with the (limited) generative literature on "scrambling", this claim is not at all obvious.

The tough counter claim to the **AAA** here is that the identity map **id** is an isomorphism relating each (a,b) pair in (8) - (10). That map does not preserve linear order: *John-i* precedes *caki-casin-ul* in (8a) but **id**(*John-i*) = *John-i* does not precede **id**(*caki-casin-ul*) = *caki-casin-ul* in (8b). But, the empirical argument would go, the relative order of arguments here is not part of Korean structure. Case marking codes grammatical function so fixing order is of no value (and so by economy isn't used). But NPs are dumb in morphologically impoverished languages (English) so the logically more complicated extrinsic ordering is invoked as a default.

There is, I think, serious prima facie support for this position. Most crucially, preserving case markers is an important (though perhaps not the only) determinant of syntactic sameness. This is seen by the fact that merely interchanging the NPs in (8) without moving their case markers results in serious ungrammaticality.

(11) a. * Caki-casin-i John-ul pinanhayssta
self- -nom John-acc criticized
He-self criticized John

b. * John-ul caki-casin-i pinanhayssta
John-acc self- -nom criticized
John criticized he-self

We are concerned here to show that it is conceptually possible to have a language which behaves as Korean appears to and which satisfies the **AAA**. Whether real Korean satisfies the **AAA** is a serious empirical question (see e.g. Lee, 1993) beyond the purview of this article[6]. Accordingly we exhibit in **2.5** a *Little Korean* in which the **AAA** is satisfied, case marking is non-trivially

structural (not just a "coding" for hierarchical structure), and the C-command relation between an A-expression and its antecedent can vary freely preserving semantic interpretation. So the syntactic and associated semantic differences between *Little English* and *Little Korean* are determined by overt morphological differences between these languages.

We will see that an important conceptual plus for our notion of *structure* is that morpheme identity, category identity, and syntactic hierarchy (e.g. *dominates*) fall out as special cases of structure -- none being derivative from the other. Thus our notion of *structure* properly generalizes the standard one.

2. Little Languages: Generalized Structure

2.1 Little English We exhibit a grammar, ENG, whose language L(ENG) = *Little English* is used to illustrate the notion *syntactically definable in G.*

ENG has some lexical *nominative NPs* (NPnom's): *john* and *he*; some lexical *accusative NPs*, NPacc's: *john, him, and himself;* some lexical *two place predicates*, P2's: *praised* and *criticized*; and some lexical *one place predicates*, P1s: *laughed* and *cried*. Rules combine P2s with NPacc's to form P1s, and P1s with NPnom's to form Ss. For each of these categories C, expressions of category C may combine in the appropriate way with *both...and...*, *either...or...*, and *neither...nor...*. to form a complex expression of the same category. L(ENG) has a simple context free grammar:

S --> NPnom + P1; P1 --> P2 + NPacc

C --> both C and C, either C or C, neither C nor C,

all C ∈ {NPnom, NPacc, P2, P1, S}

NPnom ---> john, bill, tom, bob, he
NPacc ---> john, bill, tom, bob, him, himself
P1 ---> laughed cried
P2 ---> praised, criticized

We present ENG in our *Generalized Grammar* format: First, we must define the vocabularly V = {*john, ...,cried, ...,both,...*} used. Then we define CAT, the set of categories used: {NPnom, NPacc, S, P1, P2, CONJ}. (We have added CONJ = Conjunction in order to illustrate a slight generalization). Third, we define the set LEX of lexical expressions.

In general an *expression* is treated as a pair (s,C), where C is a category symbol and s is a string built from elements of V. V^* is the set of finite strings of elements of V, and $V^* \times CAT$ is the set of *possible expressions*. In ENG (*john*,NPnom) is an element of LEX. So is (*john*,NPacc). LEX is defined (in this case, not necessarily) by listing.

Notation: If σ = (s,C) is an expression we write Cat(σ) or σ_2 for the category of σ, and we write String(σ) or σ_1 for its string part. E.g. Cat(*praised bill*,P2) = P2, String(*praised bill*,P2) = *praised bill*.

Finally, we represent the ways of combining expressions to build more complex ones by (partial) functions, called the *generating* or *structure building* functions of G. The single most important determinant of "structure" lies in specifying the domains ("structural descriptions") of these functions.

For example, given V and CAT as above for ENG, we define a function F2 which builds P1s from P2s and NPacc's as follows. The domain of F2, Dom(F2), is the set of pairs $<\sigma,\tau>$ of possible expressions which meet the condition that Cat(τ) = P2 and Cat(σ) = NPacc. The action of F2 is given by:

$$F2(\sigma,\tau) = <\tau_1 + \sigma_1, P1>$$

where we use '+' for concatenation (with or without a "space" as is convenient).

Thus F2 takes a possible expression τ of category P2 and a possible expression σ of category NPacc and yields a possible expression of category P1 whose string part is the string part of τ followed by the string part of σ.

Similarly F1 is defined to build expressions of category S from pairs of category NPnom and P1 respectively.

We illustrate the coordination rules with AND. It takes as arguments triples $<\rho,\sigma,\tau>$ of possible expressions, where ρ is simply <*and*,CONJ> and Cat(σ) = Cat(τ) is in {NPnom, NPacc, S, P1, P2}, the set of coordinable categories. The value of AND at such a triple is given by

$$AND(\rho,\sigma,\tau) = <\text{both} + \sigma_1 + \text{and} + \tau_1, \tau_2>$$

Thus AND coordinates the string parts of σ and τ in the obvious way, and the category of the resultant expression is the same as that of its conjuncts. The functions OR and NOR are defined comparably.

Finally, we define L(ENG), the *language generated by the grammar* ENG, to be the closure of LEX under the generating functions[7]. That is, L(ENG) is the set of expressions obtainable from LEX by applying the generating functions finitely many times in any way in which their structural descriptions are satisfied.

In sum: a (generalized) grammar G is a four-tuple $<V,CAT,LEX,F>$ with $LEX \subseteq V^* \times CAT$, F a set of partial functions each taking sequences of possible expressions to possible expressions, and L(G) the closure of LEX wrt F. And,

Full Expressive Power

Any collection of sets is definable by a generalized grammar[8]

FEP guarantees that our format of presentation imposes no constraints whatever on the class of languages that can be represented. In particular common X' systems are easily coded in this format (see Stabler & Keenan, 1993). (So any constraints which distinguish natural languages from other structures must be explicitly given. Nothing follows from the format of presentation itself).

2.2 Generalized Structure

We make precise the idea that two expressions have the same structure if we can match their lexical items and derivation steps in an appropriate way.

def 1 A function h from L(G) $\rightarrow$ L(G) **preserves structure** iff for all structure building functions F, (i) and (ii) below hold:

i. h preserves DomF

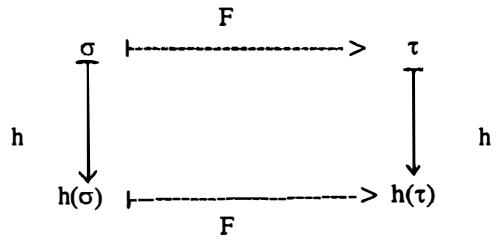
(= if a tuple σ of elements of L(G) is in DomF then $h(\sigma)$ is also in DomF)

ii. h commutes with F

(= F maps σ to τ implies F maps $h(\sigma)$ to $h(\tau)$).

(i) says that if a structure building function F can recognize (structurally analyze) a sequence σ of expressions then it can also recognize $h(\sigma)$. So h preserves the property of being analyzable by a structure building function.

(ii) says that a structure building function F treats an argument σ and its image under h "the same": the structure that F builds from $h(\sigma)$ differs from that which it builds from σ solely by what h does to σ. Pictorially, the following diagram commutes.



"Commutes" means that if we apply F to σ and then h to the result, we get the same object as if we had applied h to σ first and then F to the result. We shall often use such diagrams to express the intuitive notion "preserves structure".

notation & terminology A function h which preserves structure as above is called a *structure map* (for G). If A is a subset of L(G) then by h(A) is meant $\{h(a) \mid a \in A\}$. And if σ is a sequence, say $\sigma = <\alpha,\beta,\gamma>$, then by h(σ) is meant $<h(\alpha),h(\beta),h(\gamma)>$.

def 2 Given G with σ,τ in L(G), we say that σ is **isomorphic to** τ **in** G, written $\sigma \simeq_G \tau$, iff there is a structure map h such that $h(\sigma) = \tau$ and $h(\tau) = \sigma$.

So two expressions are isomorphic in G iff there is a structure preserving map mapping each to the other. We note that the relation *isomorphic in G* is an equivalence relation (anything is isomorphic to itself; if σ is isomorphic to τ then τ is isomorphic to σ; and if ρ is isomorphic to σ and σ is isomorphic to τ then ρ is isomorphic to τ). And our final definition

def 3 A subset P of L(G) is **syntactically identifiable** iff for all σ,

$$\sigma \in P \text{ and } \sigma \simeq_G \tau \Rightarrow \tau \in P$$

So given two expressions with the same structure, P must say yes to both, or no to both, but it cannot distinguish them. If it could then it would be relying on properties other than their syntactic structure. (Replacing 'subset' by 'relation' generalizes **def 3** to relations.)

2.3 Structure: A Linguistic Generalization

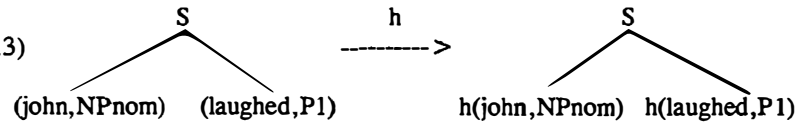
Our definitions above do not take the notion of *structure* as basic. What is basic is the notion *preserves structure*. We may think of the "structure" of an expression as those properties it must share with any expression it is isomorphic to. In a (useful) phrase:

(12) **Structure is what the Structure Preserving Functions Preserve**

NB: (12) is not circular. We have already defined what the structure preserving functions are (relative to a grammar G). We may now ask, and the answer is enlightening, just what it is that these functions may or must preserve.

i. *Structure maps must preserve the hierarchical structure of derivation trees.* E.g. let h be a structure map for ENG. Its value at $<$*john laughed*,S$>$ is given by the tree on the right in (13), where $<$*john laughed*,S$>$ is represented by the tree on the left -- a tree that represents the result of applying F1 to the pair $<$*john*,NPnom$>$, $<$*laughed*,P1$>$. The reader may show that this follows directly from the fact that h preserves Dom(F1) and commutes with it.

(13)



(NPnom | john and (john,NPnom) are notational variants).

Thus we represent the action of a structure map on a standard tree by keeping the hierarchical structure of the tree the same and applying the function to the lexical leaves. In short, structure maps preserve hierarchical structure. They may preserve much else besides.

ii. *Structure maps may preserve category.* They provably do in ENG: if $Cat(\sigma) = X$ then $Cat(h(\sigma)) = X$. So in *Little English* the category of an expression is part of its structure. It is part of what an expression must share with any expression it is isomorphic to. This fact does not follow from the definition of structure map. Not all categories must be preserved in *Little Korean* below. And current linguistic work suggests that preserving category is not a conceptually necessary part of structure. At least X' notation in part is set up to capture structural similarities among expressions of different categories.

iii. *Structure maps may preserve particular expressions* (e.g. "bound morphemes"). For example in ENG all structure maps map (*and*, CONJ) to itself. So if (*and*,CONJ) $\approx_{ENG} \tau$ then τ = (*and*, CONJ). We suggest:

Thesis The "grammatical" morphemes of a language are just the expressions which are isomorphic only to themselves[9]. In a phrase,

Grammatical morphemes are the fixed points of syntactic isomorphisms

An object x is a fixed point of a function f iff f(x) = x. f is said to *fix* x. The thesis says then that the grammatical morphemes of a language are those that an arbitrary expression must share with any expression it is isomorphic to. This is a non-trivial characterization of *grammatical morpheme*.

Note then that in L(ENG), (*both laughed and cried*,P1) does not have the same structure as (*either laughed or cried*,P1), even though their derivation trees are isomorphic qua trees, differing just on the lexical items. Reason: on the G we gave, identity of coordinating conjunction is a part of structure -- something that must be preserved by the structure preserving maps. We thus issue:

Warnings to Tree Lovers

1. Non-isomorphic expressions may have isomorphic trees
2. A fixed (unambiguous) expression may have non-isomorphic trees
3. Expressions generated in a fixed way may be isomorphic in one grammar and not isomorphic in another

(Only the first warning has been instantiated here.)

The Linguistic Generalization: Since structure is what the structure maps preserve, and morphemes and categories may be preserved, these notions may be just as "structural" as syntactic hierarchy in a given language. Thus we need not think of Korean case marking as simply coding hierarchy relations. It may be structural on its own -- that is, preserved under syntactic isomorphisms.

Finally, let us consider some sets of expressions (and relations between expressions) which are (or are not) syntactically identifiable in ENG. We note first some sets that are syntactically identifiable in all G (see Keenan and Stabler, 1991) for many more examples and more extensive discussion):

Theorem 1: Universally Identifiable Sets

Given a grammar G,

1. L(G) and $\varnothing$ are syntactically identifiable

So if σ is isomorphic to τ and σ is in L(G) then τ must also be in L(G). Thus grammatical expressions are not isomorphic to ungrammatical ones.

2. a. If A and B are identifiable so is A -- B, the set of things in A not in B

b. The union (intersection) of arbitrarily many syntactically identifiable sets is itself syntactically identifiable

(2a.b) guarantee that we can define new identifiable sets from old ones using **and**, **or** and **not**, and **universal** and **existential quantification**.

3. For $\sigma \in L(G)$ and $[\sigma]$ the set of things isomorphic to σ, $[\sigma]$ is syntactically identifiable, and no non-empty proper subset of $[\sigma]$ is identifiable

4. The *occurs in* relation is syntactically identifiable

□

Given a grammar G and a category C, write $PH_G(C)$ for the set of expressions in L(G) of category C. A $\sigma \in PH_G(C)$ is called a *phrase of category* C (*in* G).

Theorem 2: Syntactically identifiable sets in ENG 1

1. For all categories C of ENG, $PH_{ENG}(C)$ is syntactically identifiable

Thus any two isomorphic expressions in L(ENG) have the same category. Also PH_{ENG}(NPnom) -- PH_{ENG}(NPacc), the set of NPnoms that are not also NPaccs is syntactically identifiable (using **Theorem 1.2a** here).

2. The set **?ANA** of expressions of category NPacc which contain an occurrence of *himself* is not syntactically identifiable in ENG.

□

Note: **?ANA** is a reasonable first guess at a "syntactic" definition of the A-expressions of L(ENG). But it turns out that this set is not syntactically definable in terms of the grammar ENG. Reason: ENG does not allow us to distinguish *himself* from *him*. In this grammar that is like trying to distinguish *john* from *bill*. No grammatical process treats them differently.

More formally, **?ANA** is not syntactically identifiable because there is a structure map h which interchanges (*himself*,NPacc) and (*him*,NPacc). (h fixes the other lexical items and extends as a "string homomorphism" to the complex expressions). But **?ANA** is not closed under h. E.g. (*himself*,NPacc) $\in$ **?ANA** but h(*himself*,NPacc) = (*him*,NPacc) $\notin$ **?ANA**.

Thus we have pairs of expressions with the same syntactic form one of which is in **?ANA** and the other of which is not. So membership in **?ANA** cannot be predicted from the syntactic form of the expression.

2.4 A Semantic Definition of A-expression

Given a domain D of (possibly abstract) objects about which we think of ourselves as speaking on some occasion, we shall think of P1's as (minimally[10]) interpreted as subsets of D. Call such subsets (extensional) *properties*. P2's will be interpreted as binary relations over D, that is, as sets whose elements are ordered pairs of elements of D. Boolean compounds of Pn's are interpreted as the corresponding set theoretic operations of the denotations of the expressions compounded. E.g. noting interpretations in bold, the interpretation of (*laughed and praised bill*, P1) is **laugh** $\cap$ **praise bill**. That is, an object in D has the property expressed by *laughed and praised bill* iff that object lies in the interpretation of (*laugh*,P1) and also in the interpretation of (*praised bill*,P1).

Proper nouns and (*him*,NPacc) will denote (= be interpreted as) *individuals*, functions (defined below) which map properties into {**true**,**false**} and binary relation to properties (and more generally n+1 ary relations to n-ary relations).

def 4 For each $d \in D$, we define I_d by:

i. $I_d(p) =$ **true** iff $d \in p$ [all subsets p of D] and

ii. $I_d(R) = \{a \in D | \ <a,d> \in R\}$ [all binary relation R over D]

I_d is called the **individual generated by** d.

Suppose for example **john** $\in$ D and both (*john*,NPacc) and (*john*,NPnom) are interpreted as the individual generated by **john**. Then (*john laughed*, S) will be interpreted as **true** iff **john** $\in$ **laugh**, and (*criticized john*, P1) will be interpreted as the set of objects **a** $\in$ D such that $<$**a**,**john**$>$ $\in$ **criticize**. That is, (*criticized john*, P1) denotes the set of objects which stand in the **criticize** relation to **john**.

Crucially now (*himself*,NPacc) is interpreted by the function **self** below, whose domain is just the set of binary relations over D:

def 5 **self**(R) = $\{a \in D | \ <a,a,> \in R\}$

One computes then that (*john criticized himself*,S) is interpreted as **true** iff $<$**john**,**john**$>$ $\in$ **criticize**.

Boolean compounds of NPs are interpreted pointwise. E.g. (*criticized both john and himself*, P1) denotes the intersection of the denotation of (*criticized john*, P1) with that of (*criticized himself*, P1): $I_{\mathbf{john}}$(**criticize**) $\cap$ **self**(**criticize**). So an object **a** is in this set iff $<$**a**,**john**$>$ $\in$ **criticize** and $<$**a**,**a**$>$ $\in$ **criticize**.

Now, to semantically define A-expressions, we want to find a semantic property which distinguishes for example (*both john and himself*, NPacc) from (*both john and bill*, NPacc). Keenan (1987) provides just such a property (one that is independent of which object NPs the language possesses). An object NP denotation is a function mapping binary relations to properties. If the NP is not anaphoric then its function f puts an object b in the set it associates with a relation R just by checking bR, the set of things that b bears R to. It does not depend on what object b is. That is, f could not make a different decison concerning some other object b' that bore R to the same things b did. By contrast an anaphoric function can use the identity of b in making its decision.

For example, suppose that the people John praised are exactly those which Bob criticized. Then (14a,b) must have the same truth value, so *most of Jim's students* is not anaphoric. But (15a,b) may have different truth values: if John praised just Mary, Bob, Susan, and Sam. Then (15a) is false but (15b) true.

(14) a . John praised most of Jim's students
b. Bob criticized most of Jim's students

(15) a. John praised both Sam and himself
b. Bob criticized both Sam and himself

So we define:

def 6 An expression interpreted as a function mapping binary relations to properties is an *A-expression* iff in every situation its denotation f satisfies [A] and in some situations it fails [B].

[A] $aR = aS \Rightarrow a \in f(R)$ iff $a \in f(S)$

[B] $aR = bS \Rightarrow a \in f(R)$ iff $b \in f(S)$

Using this properly semantic definition of A-expression we may observe:

Theorem 3

1. (*himself*,NPacc) is the only lexical A-expression in L(ENG)
2. The set of A-expressions of L(ENG) is infinite
3. The set of A-expressions of L(ENG) is not syntactically identifiable

3.3 follows directly from 3.1 plus (*himself*,NPacc) $\sim_{ENG}$ (*him*NPacc). Also, we intended that ENG satisfy usual C-command conditions on the distribution of anaphors so that we could properly contrast it with *Little Korean*. However *bill*

asymmetrically C-commands *himself* in (16) but is not interpretable as its antecedent (nor can it be in real English).

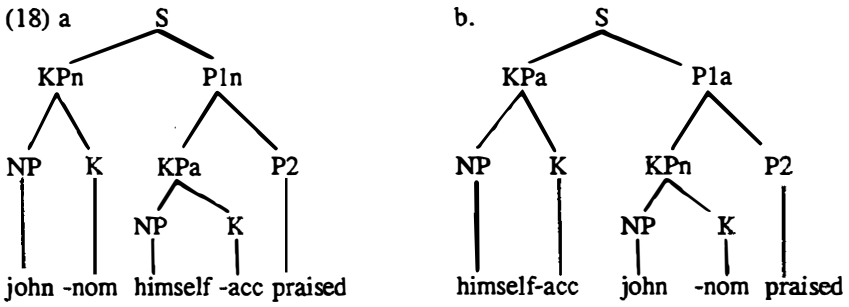
(16) (john praised either bill or both tom and himself, S)

Finally we observe that the AA (*Anaphor-Antecedent*) relation, defined below, is syntactically identifiable (a fact that does not, as we have seen with **?ANA**, follow from its apparent syntactic form)

(17) α is a *structurally possible antecedent of* β *in* γ iff $\exists\pi \in PH_{ENG}(P2)$, $F1(\alpha,F2(\beta',\pi))$ occurs in γ

2.5 Little Korean

We exhibit a grammar, KOR, which illustrates how case marking may be directly structural and how A-expressions may asymmetrically C-command their antecedents, but in which, as in L(ENG), the AA relation is syntactically identifiable. In fact in L(KOR) the (infinite) set of A-expressions is syntactically identifiable. The essence of our grammar is summarized in the two derivation trees below, plus several comments on them. We state the grammar informally, leaving details to the reader on the model for ENG.



There are only two strings of category K "Casemarker", *-nom* and *-acc*, both lexical, and there are only four strings of category NP, *john*, *bill*, *pron*, and *himself*, again all lexical. A generating function CASEMARK combines all NP strings with *-acc* to form KPa's, *accusative Case Phrases*, and it combines all NP strings except *himself* with *-nom* to form KPn's, *nominative Case Phrases*. So *himself-nom* is not a string of any category in L(KOR) and (cf real Korean) *john* and *himself* in e.g. (18b) cannot be interchanged preserving grammaticality.

F2 combines KPa's and P2s to make P1n's. It also combines KPn's and P2s to make P1a's. F1 combines P1n's with KPn's to make S's. It also combines P1a's with KPa's to make Ss. So KPs and P1s *agree* when they combine to make Ss. There are two lexical P2's as in ENG, two lexical P1n's (*laughed, cried*) and no lexical P1a's. The coordinable categories are P2, P1a, P1n, S, KPa, and KPn. Note the following structural facts re L(KOR):

Structural Facts re Little Korean

1. The categories K, NP, CONJ, and S are preserved by all structure maps. So e.g. structure preserving maps send casemarkers only to casemarkers.

2. No structure map sends anything other than (*himself*,NP) to (*himself*,NP). Thus (*himself*,NP) is isomorphic only to itself, and so is a grammatical word in L(KOR)

3. For all structure maps h, h(-nom,K) $\neq$ h(-acc,K). So we may not collapse casemarkers preserving structure. But a structure map may interchange (-nom,K) and (-acc,K) thus interchanging KPn's and KPa's, and P1n's and P1a's, provided it maps nothing to (*himself*,NP).

4. **a.** (18a) and (18b) are not isomorphic

   b. (*john-nom bill-acc praised*, S) and (*bill-nom john-acc praised*, S) are isomorphic

Fact 4 is subtle, but is provable from our definitions. An isomorphism h between (18a) and (18b) would map (*john-nom*,KPn) to (*himself-acc*,KPa), interchange (*-nom*,K) and (*-acc*,K), and map (*john*,NP) to (*himself*,NP). But $<$(*-acc*,K), (*john*,NP)$>$ $\in$ Dom(CASEMARK) and its image $<$(*-nom*,K), (*himself*,NP)$>$ under h is not, so h fails to preserve the domains of all the generating functions and so is not a structure map, contradicting the assumption.

In the semantics for L(KOR), the casemarkers (actually just (*-nom*,K), n - 1 casemarkers suffice to distinguish n KPs) are interpreted in an essential way. It turns out that (18a,b) are logically equivalent in L(KOR), notwithstanding that the A-expression and its antecedent reverse their asymmetric C-command relations.

The core of the semantics is this: proper nouns and *pron* are interpreted as individuals, and (*himself*,NP) is interpreted as **self**, as in ENG. (*-acc*,K) denotes the identity function, so KPa's behave like NPs in L(ENG). In particular they map binary relations, P2 denotations, to properties, possible P1n denotations. What is new is the interpretation of (*-nom*,K), KPn's, and P1a's.

These new denotations are all determined by the interpretation **nom** of the bound morpheme (*-nom*,K). Let us write **[R → P]** for the set of functions from the set of binary relations over the domain D into the set of properties over D. Then **nom**, by definition, takes individuals (its only possible arguments) to functions which (1) take properties to truth values, as in ENG, and (2) take binary relations to functions mapping **[R → P]** into **{true, false}**, as follows:

$$\mathbf{nom}(I_d)(R)(G) = I_d(G(R))$$

So for example, the interpretation of (*himself-acc john-nom praised*, S) is $((\mathbf{nom}(I_{\mathbf{john}}))(\mathbf{praise}))(\mathbf{self}) = I_{\mathbf{john}}(\mathbf{self}(\mathbf{praise}))$ which is just the interpretation of (*john-nom himself-acc praised*,S). Thus the fact that (18a) and (18b) are logical paraphrases is due entirely to the interpretation of the bound morphology in Little Korean, one of the obvious areas where languages admit of language particular material. And we observe:

Theorem 4

1. The set of A-expressions in L(KOR) is syntactically definable. (Note that A-expressions in Little Korean are KPa's)

2. The relation SPA_{KOR}, *is a structurally possible antecedent of*, defined below is also syntactically identifiable

def 7 $<\alpha,\beta,\gamma> \in SPA_{KOR}$ iff $\alpha \in PH_{KOR}[KPn]$ & $\exists\delta$ $F1(\alpha,F2(\beta,\delta))$ occurs in γ or $F1(\beta,F2(\alpha,\delta))$ occurs in γ.

Thus in Ls in which case marking is part of structure, it is possible to "scramble" A-expressions and their antecedents reversing C-command relations, without losing the ability to structurally identify the *Anaphor-Antecedent* relation (or even the set of A-expressions itself).

3. Condition F (or Patience Rewarded)

(Thanks separately to Dorit Ben-Shalom, Dusko Pavlovic and Ed Stabler for discussion of the points below).

Little Korean is probably as close as we get to a language that can interchange A-expressions and their antecedents freely without violating **Anaphor-Antecedent Asymmetry (AAA)**. Ultimately satisfying **AAA** is due to the fact that *himself-nom* is not a string of category KPn. If we weakened the grammar to allow it ("Irish Korean") then we would generate as Ss strings like *himself-nom cried* and *himself-nom cried and criticized bill.* These Ss would force *himself-nom* to

receive a non-anaphoric interpretation, so (18a,b) would not be logical paraphrases and **AAA** would still be satisfied.

But now that we are clear as to what counts as a syntactic isomorphism let us see what a "language" would look like that really did violate the **AAA**. The attempt will lead us to understand a condition on interpretation which runs sufficiently deep that, to my knowledge, it has never been articulated as a constraint on natural languages. This is, of course, the Fidelity condition **F**.

Part of **F** is familiar from standard considerations of compositionality: the interpretation of derived expressions should depend on the interpretation of those they are derived from (up to listable exceptions, e.g. idioms). This is, I think, conceptually necessary. If e.g. *You're standing on my foot!* could freely mean *Most unicorns like cabbage or else none do* then language would have no value as a system of representation (and derivatively of communication) and so could not be used to satisfy the conceptual-intentional functions we do in fact use it for.

But **F** will say more than this, and the more echos curiously claims in CH-92 concerning the constancy across "languages" of the computational procedure used in deriving expressions from the lexicon. The lexicon and morphology of languages do vary, but one expects the ways of building complex expressions from simpler ones to vary rather less, precisely because of their generality.

Here is a semantic analogue of these syntactic claims: Lexical items may differ semantically even when they are syntactically indistinguishable (*John/Bill*, probably *laugh/cry*, etc.). And as a consequence complex expressions which are syntactically isomorphic may be interpreted differently. *John laughed* and *Bill cried* may differ in truth value even though they are built in the same way from syntactically isomorphic parts.

But, I claim, the *procedures* we use for interpreting isomorphic expressions cannot be "radically" different. They must, in effect, be "isomorphic". But just what exactly does this mean? An example leads to the answer, **F**.

Imagine a function g which interprets lexical items of *Little English* as before but which interprets the Ss generated by F1 as in (19), where BOOL(john,bill) is the set of NPnoms buildable from {(john,NPnom), (bill,NPnom)} with *both...and...*, *either...or...*,and *neither...nor...* and $-$ is boolean complement.

$$(19)\quad g(F1(\sigma,\tau)) = \begin{cases} g(\sigma)(g(\tau)) \text{ if } \sigma \notin \text{BOOL(john,bill)} \\ -(g(\sigma)(g(\tau)) \text{ otherwise} \end{cases}$$

So e.g. *both john and tom laughed* is true iff John laughed and Tom laughed. But *both john and bill laughed* is true iff either John didn't laugh or Bill didn't laugh. This is outrageous. What we say on this "semantics" about John and Tom is not what we say about John and Bill in the isomorphic sentence. What is wrong here is that in deciding how to interpret an expression, g refers to a set BOOL(john,bill) which is not syntactically identifiable. (e.g. (*john*,NPnom) is in it but (*bob*,NPnom) is not, even though these two NPs are isomorphic).

We now set up the general principle F which will rule out perverse attempts at interpretation like g.

Let L = L(G) be given. An *interpretation* of L is simply an arbitrary function μ with L included in its domain. So such a μ assigns values to the expressions of L. No assumptions are made concerning μ[L], the set of values of μ. μ[L] might in particular be some appropriately coded form of "conceptual-intentional" structure. Here we shall neutrally refer to μ[L] simply as the "μ world".

Observe though that where h is a structure map for L the notation μ(h) makes sense. We think here of h as the set of pairs $<\delta,h(\delta)>$ for each expression δ in L(G). Then μ(h) is just the set of pairs $<\mu(\delta),\mu(h(\delta))>$. And for F a generating function of L, we write μ(F) for the set of pairs $<\mu(\sigma),\mu(F(\sigma))>$, where σ is a sequence of elements of L(G) in the domain of F.

Now not just any way of interpreting a language is a conceptually possible *semantic* interpretation. We require that semantic interpretations μ are *faithful* to the syntax of the language, as per (1) repeated as (20) below:

(20) **F**

Semantic interpretations of a language L are faithful to the syntax of L

def 8 An interpretation μ of a language L(G) is *faithful* to the syntax of L(G) iff μ satisfies **1** and **2**:

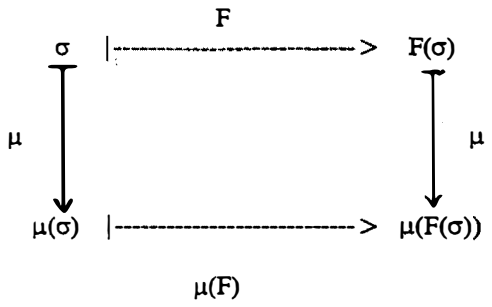
1. For all structure building functions F of G, μ(F) is a (structure building) function in μ[L]

2. For all structure maps h of G, μ(h) is a (structure preserving) function in μ[L]

Discussion

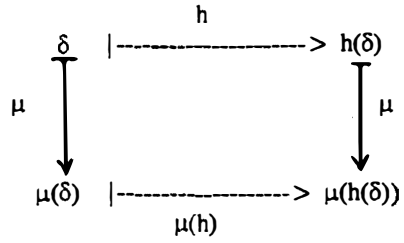
The basic idea is that a semantic interpretation of a language not only interprets the expressions, it respects the structure of L -- in particular it respects the relation between a complex expression and what it is derived from, and it respect the sameness of structure relation between different expressions built in the same ways. In more detail:

1. **F(1)** says that if F builds τ from σ, $F(\sigma) = \tau$, then $\mu(F)$ builds $\mu(\tau)$ from $\mu(\sigma)$, $(\mu F)(\mu\sigma) = \mu(\tau)$. That is, the following diagram commutes (a fact that is guaranteed as long as $\mu(F)$ as given above is a function):



Note that **F(1)** is just standard compositionality: if an expression τ is derived as a function F of a tuple σ of expressions then the interpretation $\mu(\tau)$ of τ is a function of the interpretation $\mu(\sigma)$ of σ. That function of course may depend on how τ was built from σ, that is, on F.

2. **F(2)** is the condition that is properly new. It guarantees that expressions with the same syntactic structure have the same semantic structure. Specifically, the following diagram commutes:



We might note that as long as $\mu(h)$ is a function then it preserves structure in the μ world: that is, it preserves the domains of the structure building functions $\mu(F)$ and it commutes with them[11].

3. Let us see that **F(2)** is independent of **F(1)**. Let μ be a function whose domain is L(ENG) = *Little English* satisfying:

$\mu(\textit{laughed},\text{P1}) = \{\textbf{tom},\textbf{bob},\textbf{bill}\}$, $\mu(\textit{tom},\text{NPnom}) = \textbf{tom}$, etc. and

$$\mu(F1(\sigma,\tau)) = \begin{cases} \mu(\sigma)(\mu(\tau)) \text{ if } \sigma \notin \text{BOOL}(<\text{john},\text{NPnom}>,<\text{bill},\text{NPnom}>) \\ -[\mu(\sigma)(\mu(\tau))] \quad \text{otherwise} \end{cases}$$

So μ(F1) maps a pair $<\mu\sigma,\mu\tau>$ to $\mu(F1(<\sigma,\tau>)$, as above. μ(F2), μ(AND), etc. are defined similarly. Thus **F(1)** is satisfied.

Let h be a structure map which interchanges (*tom*,NPnom) and (*bill*,NPnom) and maps each other lexical expression to itself. (There is such an h). Then, omitting category indices for simplicity, and writing 1 for **true** and 0 for **false**, suppose, leading to a contradiction, that μ is a function. Then,

$\mu(h)(1) = \mu(h)(\mu(\text{tom})(\mu(\text{laughed}))) = \mu(h)(\mu(\text{tom laughed}))$

$= \mu(h(\text{tom laughed})) = \mu(h(\text{tom}),h(\text{laughed})) = \mu(\text{bill laughed})$

$= -[\mu(\text{bill})(\mu(\text{laughed})] = -(1) = 0.$

But also,

$\mu(h)(1) = \mu(h)(\mu(\text{bob})(\mu(\text{laughed})) = \mu(h)(\mu(\text{bob laughed}))$

$= \mu(h(\text{bob laughed})) = \mu(h(\text{bob}),h(\text{laughed})) = \mu(\text{bob laughed})$
$= \mu(\text{bob})(\mu(\text{laughed})) = 1$

So $\mu(h)(1) = 1$ and $\mu(h)(1) = 0$, contradiction. So **F(2)** fails. ♥

4. Equally a simple language shows that **F(2)** does not entail **F(1)**, and thus the two conditions are independent[12].

4. Condition F implies Anaphor-Antecedent Asymmetry

We attempt to construct a language, **L-SYM**, in which **AAA** is violated.

(21) L violates **Anaphor Antecedent Asymmetry** iff L presents isomorphic transitive Ss which are logically equivalent in which the anaphor in one is mapped by the isomorphism to the antecedent in the second, and vice versa.

L-SYM is given schematically below:

P1: laughed, cried P2: praised, criticized

NP1: john, bill, he NP2.1: john, bill, he NP2.2: john, bill, himself

F1: $(s,NP1),(t,P1) = (s+t,S)$

F2: $Dom(F2) = \{<\sigma,\tau,\pi> \mid \sigma_2,\tau_2 \in \{NP2.1, NP2.2\} \ \& \ \sigma_2 \neq \tau_2 \ \& \ \pi_2 = P2\}$

$F2(\sigma,\tau,\pi) = (\sigma_1 + \tau_1 + \pi_1, S)$

Coordination rules as usual, just combine expressions of the same category.

fact There is a (bijective) structure map h which maps all expressions of category NP2.1 to ones of category NP2.2 and vice versa. h interchanges (*john*,NP2.1) and (*himself*,NP2.2) and fixes all P2s. So (22a,b) are isomorphic in L-SYM.

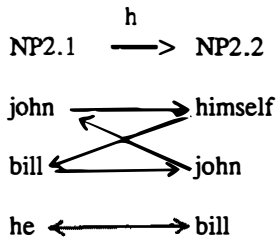
(22) a. (john himself praised,S) b. (himself john praised,S)

Moreover the interpretations μ we design make them logically equivalent. These μ interpret lexical items, Ss built by F1, and boolean compounds as expected. Only the interpretation of expressions built by F2 is tricky.

$$\mu(F2(\sigma,\tau,\pi)) = \begin{cases} \mu(\sigma)(\mu(\tau(\mu(\pi)))) \text{ if } (\sigma_1,NP2.1) \in L(G) \text{ and} \\ \mu(\tau)(\mu(\sigma(\mu(\pi)))) \text{ otherwise} \end{cases}$$

Thus a μ looks at an expression (x + y + praised,S), checks if x is a string of category NP2.1, in which case it interprets it as applying last. If x is not such a string then the interpretation of (y,NP2.1) applies last and μ(x,NP2.2) maps the binary relation to a property. But these μ then must be able to tell whether an expression is in PH_{SYM}[NP2.1], and this set is not syntactically identifiable (since e.g. (*john*,NP2.1) is in it but (*himself*,NP2.2) is not). So it should be the case, and is, that our μ in general fail F(2).

To see this, let μ be a function with domain L-SYM, and let μ identify <*john*,NP2.1> and <*john*,NP2.2> (and ditto for <bill, ...>. (Actually this requirment is almost forced since μ is required to be a function and so must assign just one value to e.g. <*john bill praised*,S> though this latter has two sources, one with <*john*, NP2.1> and the other with <*john*, NP2.2>. Call this common interpretation **john**. Let h be the identify on LEX except for the stipulations given below in which h permutes the lexical elements of PH_{SYM}[NP2.1] and PH_{SYM}[NP2.2].



Then:

1. μ(h)(**john**) = μ(h)(μ<john,NP2.2>) = μ(h(<john,NP2.2>)

= μ(<john,NP2.1>) = **john**

so <**john**,**john**> ∈ μ(h) and

2. μ(h)(**john**) = μ(h)(μ(<john, NP2.1>)

= μ(h(john,NP2.1) = μ(<himself,NP2.2>) = **self**

so <**john**,**self**> ∈ μ(h), so μ(h) is not a function, violating **F(2)**. □

And in general if L does not syntactically distinguish anaphors and non-anaphors then interpreting functions can not distinguish between them satisfying **F(2)**. But interpreting functions must distinguish between them in order to know which to apply second in minimal transitive Ss. Thus an L which does not distinguish between anaphors and non-anaphors will not admit of a faithful interpreting function.

Thus does **Fidelity** predict **Anaphor Antecedent Asymmetry.**

Footnotes

* This work was supported in part by the Social Sciences and Humanities Research Council of Canada. My thanks also to Jim Lambek and the many members of the McGill Mathematics and Statistics department to whom I presented this work.

1. These would correspond to the LFs in the format of CH-92.

2. *Caveat*: **F** makes a claim that should be assessed on its own merits, and not according to whether it captures anything intended in CH-92 (which I think it does not). But reading CH-92 after presenting this principle I could not help but be struck by the fact that **F** has many of the generality properties Chomsky associates with "(virtual) conceptual necessity". It falls just short, it seems, of logical necessity. Certainly the counterexample to **F** constructed in 3 is linguistically unnatural. If further work supports the truth of **F** but places it outside the "(virtual) conceptual necessities" countenanced within the Minimalist Framework, we shall at least have learned something about the nature of those necessities.

3. While most work on anaphora in generative grammar is consistent with the claim that the set of A-expressions (and the *anaphor-antecedent* relation) are syntactically characterizable, that work generally lacks a semantic definition of these notions and hence is conceptually unprepared to falsify these claims. Note that the items which (on our view) are defined as A-expressions are syntactic objects (expressions), but they are defined as those expressions which satisfy some semantic condition. So it makes sense to ask if the grammar G provides the syntactic tools to yield a syntactic definition of the same set. The case is quite analogous to whether the set of logically true sentences in one or another mathematical language can be syntactically defined. (And the answer is that it depends on the language -- for some it can be, for others it cannot).

4. This usage is more widespread in England than is apparent from the literature. I have also found American English speakers of Irish descent whose home communities maintain this usage.

5. For presentational purposes I have, reluctantly, followed the Anglo-centric bias implicit in the term 'scrambling' which takes the word order freedom in Korean (Chinese, Bengali, Japanese, Malayalam) as a phenomenon which needs explaining. An alternative view is that their word order freedom is simply the consequence of not having applied FIX ORDER. There is no gain to fixing the order of arguments, as the independently present morphology provides the desired information. But applying FIX ORDER yields a benefit in morphologically impoverished languages, as their NPs are dumb: they cannot tell us what their

function is. So what we explain on this view is word order rigidity in English, not its freedom in Korean.

6. It is worth noting however that the easy tests that discriminate e.g. *John likes himself* and *Himself John likes* in English do not carry over to Korean. The reflexive first order in English is largely limited to root clauses, does not feed extraction (**the man that himself likes*, **Who does himself like?*) and does not admit of a free choice of antecedent (**?Himself noone likes*). But in Korean (and Bengali) reflexive first order occurs easily in subordinate clauses, (i).

i. a. Mary-ka caki-casin-ul hoyhap-eyse pinanhayssta
Mary-nom self- -acc meeting-at criticized
Mary criticized herself at the meeting

b. caki-casin-ul Mary-ka pinanhayssten hoyhap
self- -acc Mary-nom criticized meeting
the meeting at which Mary criticized herself

Also the major way of focussing on an "object" is not to "front" it, but rather to mark it with a "Topic" postposition, in the preverbal position, (ii).

ii. Linda-ka Mary-nun pinanhayssta
Linda-nom Mary-top criticized
Linda criticized MARY (as opposed to someone else)

So again in Korean morphology is used where word order and presumably hierarchy variation are used in English.

7. Formally, $L(G) =_{df} \bigcap\{B \subseteq V^* \times CAT \mid LEX \subseteq B \ \& \ B \text{ is closed wrt each } F \in \mathcal{F}\}$

8. More precisely, let $\{K_i\}_{i \in I}$ be an indexed family of sets. Set $G_I = \langle \bigcup_i K_i, I, \{\langle s,i\rangle \mid s \in K_i\}, \varnothing\rangle$. Then $L(G_I) = LEX$ and for each category $i \in I$, the set of strings of category i is exactly K_i.

9. The Thesis gets a basic idea right here, but it needs be generalized. One approach is elaborated in Stabler & Keenan (1993).

10. A richer semantics is ultimately needed for natural language, but the minimalist extensional representation here is sufficient to capture core facts regarding binding and quantifier scope.

11. proof:

1. $\mu(h)$ preserves $Dom(\mu(F))$: Let $\mu(\delta) \in Dom(\mu(F))$, so $\delta \in Dom(F)$. But then $h(\delta) \in Dom(F)$ since h is a structure map and so preserves Dom(F). But then $\mu(h)(\mu(\delta)) = \mu(h(\delta)) \in Dom(\mu(F))$.

2. $\mu(h)$ commutes with $\mu(F)$:

$$\begin{aligned}
\mu(h)(\mu(F)(\mu\delta)) &= \mu(h)(\mu(F\delta)), && \text{def } \mu(F)\\
&= \mu(h(F\delta)) && \text{def } \mu(h)\\
&= \mu(F(h(\delta))) && \text{h commutes with F}\\
&= \mu(F)(\mu(h(\delta)) && \text{def } \mu(F)\\
&= (\mu F)(\mu(h)(\mu d)) && \text{def } \mu(h)
\end{aligned}$$

□

12. Observe first that given any grammar G, a structure map h is the identity function on L(G) iff for all $\delta \in LEX$, $h(\delta) = \delta$. (Proof by recursion: h maps each δ in LEX to itself, and if h maps a tuple σ in Dom(F) to itself then h maps $F(\sigma)$ to itself since $h(F(\sigma)) = F(h(\sigma)) = F(\sigma)$).

Now we construct a simple G in which the only structure map is **id**, the identity function. Since $\mathbf{id} = \{<\delta,\delta> | \delta \in L(G)\}$ then for *any* map μ whose domain includes L(G) , $\mu(\mathbf{id}) = \{<\mu\delta,\mu\delta> | \delta \in L(G)\}$ is a function. That is, any map from L will send each structure map to a (structure preserving) function.

Let G have three generating functions: F, H, and K, with domains:

Dom(F)	Dom(H)	Dom(K)
<*john*,NP>	<*bill*,NP>	<*john*,NP>
<*bill*,NP>	<*sam*,NP>	<*sam*,NP>

Where ϕ is any of these functions and δ any element of its domain, $\phi(\delta) =_{df} <\delta_1 + \textit{laughed}, S>$.

Let μ be given as follows: μ maps all NPs to a fixed object **a**, and all other expressions σ to String(σ). So $\mu(F)$ contains $<\mu(\textit{john},NP), \mu(F(\textit{john}, NP)> = <a,\textit{john laughed}>$ and also $<\mu(\textit{bill},NP),\mu(F(\textit{bill},NP))> = <a,\textit{bill laughed}>$. So $\mu(F)$ is not a function. □

References

1. Chomsky, N. (1992) "A Minimalist Program for Linguistic Theory" *MIT Occasional Papers in Linguistics*, No. 1, Dept. of Linguistics and Philosophy, MIT Cambridge, MA 02139.

2. Keenan, E. L. (1987) "Semantic Case Theory" In *Proceedings of the Sixth Amsterdam Colloquium* J. Groenendijk, M. Stokhof, and F. Veltman (eds.), ITLI (Univ. of Amsterdam), 1987. Reprinted in: *Semantics and Contextual Expression* R. Bartsch, J. van Benthem, and P. van Emde Boas (eds.), *Foris* 1989

3. Keenan, E. L. (1988) "On Semantics and the Binding Theory" In *Explaining Language Universals*, J. Hawkins (ed.), Basil Blackwell

4. Keenan, E. L. (1991) "Anaphora Invariants and Language Universals" In *Proceedings of WCCFL* X, Dawn Bates (ed.), Stanford Linguistics Association, Stanford University.

5. Keenan, E.L. and E. P. Stabler (1991) "Language Invariants" in *Proc. of the Eighth Amsterdam Colloquium* Paul Dekker & Martin Stokhof (eds) Institute for Logic, Language and Computation University of Amsterdam, 309 - 329.

6. Lee, H. (1993) "Binding Theory and the Principle of Referential Autonomy" to appear in *Proc. of WCCFL XII*, CSLI, Stanford CA.

7. Stabler, E. and E. Keenan (1993) "Syntactically Definable Sets" presented at the joint meetings of the Linguistic Association of America and the Association for Symbolic Logic, Ohio State University.