NESTED COOPER STORAGE: THE PROPER TREATMENT OF QUANTIFICATION IN ORDINARY NOUN PHRASES

William R. Keller

1987

17012148

Cognitive Science Research Paper

Serial No. CSRP 073

Cognitive Studies Programme
The University of Sussex,
Brighton, BN1 9QN

O 73

Nested Cooper Storage: The Proper Treatment of Quantification in Ordinary Noun Phrases

William R. Keller.

University of Sussex, Cognitive Studies Programme,
Falmer, Brighton BN1 9QN, England.

January 1987.

ABSTRACT

A technique devised by the linguist Robin Cooper provides a means of describing quantifier scope without the need for otherwise unmotivated syntactic ambiguity. It can be shown that certain complex noun phrases may exhibit quantifier scope ambiguity, but that Cooper's strategy tends to overgenerate in such cases. A consideration of the reasons for this suggests a modification of Cooper's theory which overcomes the problem. The result is a principled approach to quantifier scope generation which is more restrictive than 'naive' techniques in that it predicts fewer readings. The strategy offers computational benefits which may make it more suitable for implementation in natural language processing systems.

January 21, 1987

Nested Cooper Storage: The Proper Treatment of Quantification in Ordinary Noun Phrases

William R. Keller.

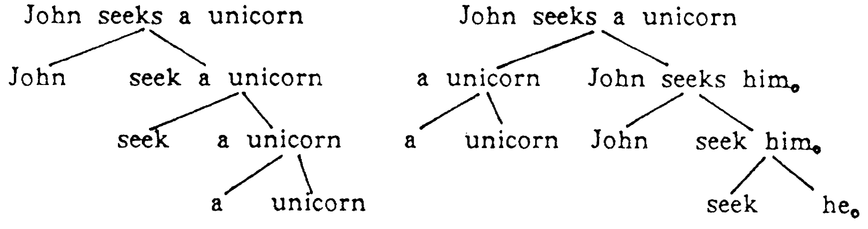
University of Sussex, Cognitive Studies Programme,
Falmer, Brighton BN1 9QN, England.
January 1987.

Keywords and Phrases: syntax, semantics, grammar, parse tree, structural description, model theory, intensional logic, quantification, quantifier scope, ambiguity, computational linguistics, natural language processing.

1. *Introduction*

Certain kinds of ambiguity in natural languages arise from the interaction of quantified noun phrases. According to the grammar of PTQ [Montague 1974b], a sentence such as *John seeks a unicorn* has two interpretations depending soley on the scope assigned to the quantified noun phrase *a unicorn*. In fact the sentence is regarded as syntactically ambiguous, having the two distinct analysis trees shown below. The analysis in (1a) is intended to represent the so-called narrow-scope, or *de dicto* reading on which John has no particular unicorn in mind, whilst (1b) corresponds to the wide-scope, or *de re* interpretation.

(1)a. b.



The advantage of disambiguating natural language in this way is that it allows semantic interpretation to be treated as a function: a one-one mapping from struc-

---

I would like to thank Gerald Gazdar, Stuart Shieber and Hans Uszkoreit for comments and discussions relating to this paper.

tural descriptions to objects in a suitable domain of discourse [Montague 1974a]. The signal disadvantage is that structural ambiguity may not always be well-motivated on syntactic grounds.

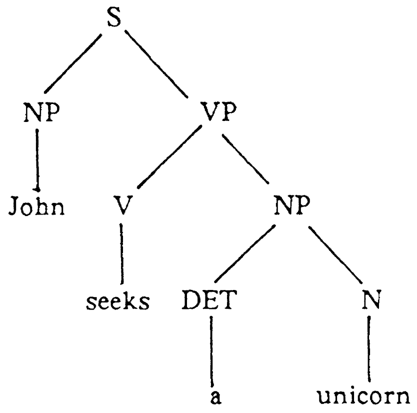
The linguist Robin Cooper was able to show that the need for such syntactic disambiguation could be avoided without losing the essence of Montague's approach [Cooper 1975]. Cooper chose to define semantic interpretation not as a function, but as a relation between structures and meanings. Following PTQ, in [Cooper 1975] this relation is defined in two stages: structural descriptions are first translated into one or more expressions of intensional logic (IL) and these latter are then given a fairly standard model-theoretic interpretation. Cooper has since provided a more comprehensive version of his theory which rejects translation in favour of direct interpretation of phrase structure [Cooper 1983]. It is arguable that direct interpretation emphasizes that what is being defined is a meaning relation. However, for ease of exposition the presentation of Cooper's strategy given in this paper will be more in the spirit of his earlier work. We shall thus be concerned with a technique for translating structural descriptions into IL. It is important to note that the criticisms of Cooper's strategy and the modifications subsequently proposed carry over just as well to the later theory.

In the rest of this paper Cooper's strategy is looked at in more detail. Particular attention is paid to the technique used to relate parse trees to logical expressions, and this is outlined in the next section. Following a consideration of quantifier scope ambiguity arising in noun phrases, in section 3 it will be demonstrated that Cooper's strategy can give rise to incorrect results. The reasons for this are discussed and a solution proposed in section 4. Finally , in section 5 the advantages of implementing the revised strategy in natural language processing systems are briefly considered.

2. *Cooper Storage.*

Coopers's strategy will be illustrated informally with a simple example, the translation of *John seeks a unicorn.* As a basis for translation, the syntax tree in (2) should suffice.

(2)



The adopted strategy will be to work from the leaves to the root, assigning interpretations to successively higher nodes. In accordance with the principle of semantic compositionality, the interpretation associated with some given node in the tree will always be a function of the interpretations of its daughters. An interpretation of a phrase structure node is itself a n-place sequence, the first element of which is a suitable denotation for the node in question: in this paper an expression of IL. Subsequent elements, if any, represent stored *binding operators.* Each binding operator is derived from the denotation of a previously encountered NP node and carries a unique index. Indexes themselves refer to free variables occurring in the denotation assigned to the node.

For example, the interpretation for the DET node will be the one-place sequence <a'>. The first, and only element of this sequence is the denotation of the determiner *a* and there are no binding operators in store. Similarly, the interpretation associated with the N node in (2) will be the one-place sequence <unicorn'>.

To obtain an interpretation for the NP *a unicorn* the interpretations of the DET and N nodes must now be combined. This is achieved by combining the denotations for the nodes after the fashion of PTQ, and forming a new sequence with the resulting expression as first element. Any binding operators in store at either the DET or N nodes are also to be included in the interpretation. In this case there are none, and the result is therefore a new one-place sequence $\langle a'(unicorn') \rangle$.

Once the V node has been assigned the interpretation $\langle seek' \rangle$, the VP node may be dealt with. At this point according to Cooper, the option of storing the NP translation also arises. Ignoring this alternative for the moment, the result of combining the V and NP interpretations will be $\langle seek'({}^\wedge a'(unicorn')) \rangle$. Next, the NP *John* is assigned the interpretation $\langle \lambda P.P\{j\} \rangle$ which may be combined with that of the VP to yield $\langle seek'({}^\wedge a'(unicorn'))(j) \rangle$ as the interpretation associated with the S node. Since there are no binding operators in store, the denotation $seek'({}^\wedge a'(unicorn'))(j)$ may be regarded as a valid translation for the whole tree. Note that it corresponds to the narrow-scope, or *de dicto* reading of the sentence.

To generate the *de re* reading translation proceeds just as before until the option of storing the translation of *a unicorn* arises. At this point the NP-Storage rule is invoked:

NP-Storage:

> If α is an NP node with meaning (i.e. denotation) α', then the sequence $\langle \lambda P.P\{x_i\},[\alpha']_i \rangle$, for some unique index i, is a valid interpretation for that node. (The variable x_i will be called an **address variable**).

Application of this rule associates with the NP node an additional interpretation $\langle \lambda P.P\{x_0\},[a'(unicorn')]_0 \rangle$, where x_0 is the chosen address variable. Combining this new sequence with that of the V node $\langle seek' \rangle$, gives rise to a further interpretation for the whole VP $\langle seek'({}^\wedge \lambda P.P\{x_0\}),[a'(unicorn')]_0 \rangle$. It should be noted that the stored binding operator derived from the NP translation is carried up the tree to the VP node. Similarly, combining the NP interpretation with that of

John, $<\lambda P.P\{j\}>$, yields a new interpretation for the sentence as a whole $<seek'({}^\wedge\lambda P.P\{x_0\})(j),[a'(unicorn')]_0>$.

This time there is a binding operator in store corresponding to the meaning of the NP node. Furthermore the partial translation $seek'({}^\wedge\lambda P.P\{x_0\})(j)$ contains a free occurrence of the address variable x_0. In order to complete the translation it is necessary to retrieve the binding operator by a process which will here be named **Storage Retrieval**. An appropriate retrieval rule for S nodes is given below.

S-Retrieval:

> If α is an S node and the sequence $<\phi, \sigma^1,[NP']_i, \sigma^2>$ is an interpretation for α, then the sequence $<NP'({}^\wedge\lambda x_i.\phi), \sigma^1\sigma^2>$ is also an interpretation for α. (Where σ^1 and σ^2 are sequences of stored binding operators and ϕ is an expression of IL)

Applying S-Retrieval yields the one-place sequence $<a'(unicorn')({}^\wedge\lambda x_0.seek'({}^\wedge\lambda P.P\{x_0\})(j))>$. The denotation thus constitutes a further translation for the whole tree. Assuming the usual PTQ translation for the determiner *a* it may be reduced to the logically equivalent (3), which corresponds to the wide-scope, or *de re* reading as required.

(3) $Ex[unicorn'(x)\&seek'({}^\wedge\lambda P.P\{x\})(j)]$

In this way both interpretations of the sentence *John seeks a unicorn* may be obtained without the need for otherwise unmotivated syntactic ambiguity.

The technique may also be applied to cases where ambiguity arises from the interaction of two or more quantified NP's. For example, one possible representation for *every man finds a unicorn* is shown below.

(4) $<find'({}^\wedge\lambda P.P\{x_0\})(x_1),[every'(man')]_0,[a'(unicorn')]_1>$

In this instance there are two stored binding operators, and two address variables (x_0 and x_1) occur unbound in the partial translation. To empty the store it is therefore necessary to apply the S-Retrieval rule twice. But now note that there are two ways of going about this. Either the binding operator corresponding to *every man* may first be retrieved, followed by that of *a unicorn,* or alternatively the opposite ordering may be chosen. Whichever way. the result will be a one-place sequence, and in both cases the expression of IL will be a valid translation for the sentence. The two translations will however differ in the relative scope assignments to the NP denotations every*(xnanO and a'(unicorn'), and accordingly represent different model-theoretic interpretations.

3. *Complex Noun Phrases and Scope Ambiguity.*

Noun, phrases often exhibit a fair degree of structure involving modifiers such as adjectival phrases and relative clauses, and complements including prepositional phrases, non-finite verb phrases and *that* clauses [Chomsky 1970], [Gazdar *et al* 1985]. It is interesting to note that where the structure is rich enough, quantifier scope ambiguities may show up apparently similar to those arising in clauses. The NP's italicised in (5a) to (5d) exhibit such ambiguity.

(5)
a. *An agent of every company* arrived
b. They disqualified *a player belonging to every team.*
c. *Every attempt to find a unicorn* has failed miserably.
d. Fortunately, *every recommendation that a hospital be demolished* was ignored.

The *NP* italicised in (5a) involves a PP acting as a complement of the noun *agent.* There are clearly two distinct readings: one on which there is a single agent, as well as the perhaps more natural interpretation involving many agents, one for each company. The ambiguity presumably arises from the scope given to the quantified NP *every compony.* Likewise in (5b) there may be a single player or many. In this case a quantified NP *every team* occurs as a constituent of a modifying VP.

The examples in (5c) and (5d) illustrate cases involving VP and S complements respectively. In (5c) both *de dicto* (no particular unicorn) and *de re* (one, alternatively many unicorns) readings seem to be available. In this respect there is a striking similarity between (5c) and the sentence *John seeks a unicorn.* The italicised NP of *{5d)* evinces the same kind of semantic ambiguity.

The foregoing examples indicate a rather general phenomenon operating across a variety of structures. In each case it appears that an embedded quantified NP may plausibly be given overall wide-scope with respect to the entire NP. Admittedly, for many NP's similar to those above, narrow-scope readings often seem to be preferred. On the whole it appears that examples involving PP modifiers or complements give rise to wide-scope readings most readily. It is a little harder to find convincing examples where VP modification or complementation occurs, and quite difficult in the case of S complements and relative clauses. Why this should be so is not immediately obvious and merits further attention, but for the present it will be accepted that with more or less ease, wide-scope readings may be obtained in sufficiently many cases to justify some kind of general treatment.

Consider now the examples in (6) below.

(6)

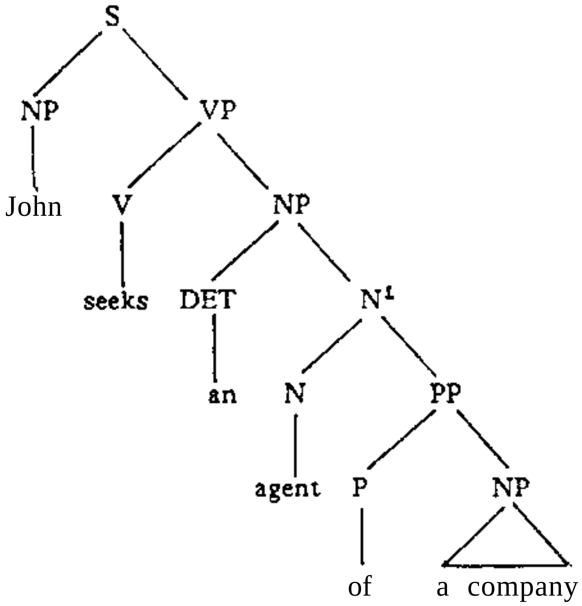
a. John seeks *an agent of a company.*

b. John seeks *a company agent.*

Given that the NP's italicised in (6a) and (6b) might initially appear synonymous, it is rather surprising to find that sentence (6a) may be read in three different ways. Not only can the object term be given *de dicto* and *de re* interpretations as in (6b), but additionally *a company* may be read wide of *seek*, whilst *an agent (of it)* retains narrow-scope. This might be paraphrased: 'There is a particular company such that John seeks some or other agent of that company'. Here there is evidence that the scope assigned to an embedded NP is relatively independent of that given to the larger NP. This fact must be properly accounted for by any proposed translation strategy.

Suppose that Cooper's strategy is adopted to induce an appropriate set of translations for (6a). It can first be assumed that the sentence has the structure shown below.

(7)



By analogy with the translation of *John seeks a unicorn* given in section 2, it i: immediately clear that a *de dido* reading for *an agent of a company* may be obtainec without invoking the NP-Storage rule. Similarly, if it is chosen to store the whole object NP. then S-Retrieval will lead to the *de re* interpretation. In order to obtain the third reading mentioned above it is necessary to store the embedded NP *a company* as in (8) [1].

(8) <atagent*CXPJP{x_o}))IatcompanyO]$_0$ >

Translation now continues as for the narrow-scope reading, with the result that the S node has the following sequence assigned to it.

(9) <seek*Ca*(agent'CAP&{x_o }))Xj)Xatcompany)]$_o$ >

This completed, the stored binding operator may be retrieved giving *a company* wide-scope, whilst *an agent(of it)* remains narrow with respect to *seek.*

So far then, it appears that Cooper Storage works, but closer inspection reveals that

---

[1]. The analysis of relational nouns assumed here is that of [Gazdar *et al* 1985]. A relational loun such aj *agent* will (logically speaking) be of type < **<s,Type{NP)>$_r$<c,t>** >, that is, **a** 'unction from the intensions of term phrases to /st,t? of individuals. It should further be noted :hat the preposition *of* is to be treated as the identity function on term phrases.

there is a further reading for (6a) licenced by the storage mechanism. This involves storing both the denotation of *a company* as well as that of the larger NP of which it forms a part.

Starting with the interpretation previously obtained for *an agent of a company* shown in (8) above, NP-Storage is now invoked for a second time to yield the further three-place sequence shown in (10).

(10) $<\lambda Q.Q\{x_1\},[a'(agent'({}^\wedge\lambda P.P\{x_0\}))]_1,[a'(company')]_0>$

Combining this with the meaning of the V node, and the result with that of *John* produces a new interpretation to be associated with the sentence as a whole.

(11) $<seek'({}^\wedge\lambda Q.Q\{x_1\})(j),[a'(agent'({}^\wedge\lambda P.P\{x_0\}))]_1,[a'(company')]_0>$

The resulting sequence has two binding operators in store, but it is notable that the expression $seek'({}^\wedge\lambda Q.Q\{x_1\})(j)$ contains only one free address variable (x_1). The second address variable x_0, occurs free in the binding operator $[a'(agent'({}^\wedge\lambda P.P\{x_0\}))]_1$.

An immediate consequence of this is that the S-Retrieval rule is no longer guaranteed to produce sensible results. To see that this is so, it may be noted that if $[a'(company')]_0$ is retrieved first, then the resulting translation will be as shown in (12a). Substituting the PTQ translation for the determiner *a* and simplifying, this reduces to the logically equivalent (12b).

(12)
a. $a'(agent'({}^\wedge\lambda P.P\{x_0\}))({}^\wedge\lambda x_1.a'(company')({}^\wedge\lambda x_0.seek'({}^\wedge\lambda Q.Q\{x_1\})(j)))$
b. $Ex[agent'({}^\wedge\lambda P.P\{x_0\})(x)\&Ey[company'(y)\&seek'({}^\wedge\lambda Q.Q\{x\})(j)]]$

Roughly speaking, the problem with this 'translation' is that it still contains a free occurrence of the address variable x_0. This is since the retrieval of the translation of *a company* made it impossible to bind the variable with the lambda abstraction operator as would normally happen.

It is easily seen that performing retrieval in the opposite order does not lead to the same problem, since both address variables become bound. The translation and its simplified equivalent are shown in (13a) and (13b) respectively [2].

(13)

a. a*companyOCAx_0 .a'Ugent'CAP J>{x_0))X~A x_x .seek'CAQ.QU$_x$ }Xj)))

b. Ex[company'(x)&Ey[agent'(^λP.P{x})(y)&seek'(^λQ.Q{y})(j)]]

4. *Nested Cooper Storage.*

From the foregoing discussion it should be clear that Cooper Storage depends for its success on the way in which address variables are utilised during translation. Intuitively, each address variable acts as a kind of semantic place-holder, relating an indexed binding operator to the role it will eventually play in the final translation. To illustrate, the representation for *every man finds a unicorn* is reproduced in (14) below. Inspection of the address variables reveals that the 'something doing the finding' corresponds in some sense to the stored item every*dnan$^>$), whilst the 'thing being found' corresponds to a*(iinicoroO.

(14) <fiLndTAP.P{x_0}Xx Jlevery'dnanOJoIa^unicornOli >

The interpretation of *John seeks an agent of a company,* which is reproduced in (15), may be read in a similar fashion. Here the 'thing being sought' corresponds to a*(agent'CAPJP{x_0})), but notice that the role of accompany*), indicated by the address variable x_0, is 'that thing the agent is an agent of. Importantly, it does not play a (direct) role in the expression seek'CAQ.Qfx^Xj) as evidenced by the absence of the address variable x_0.

(15) <seek'(^λQ.Q{x_1})(j),[a'(agent'(^λP.P{x_0}))]$_1$,[a'(company')]$_0$>

The problem thus seems to be that the S-Retrieval rule is insensitive to the precise roles of the stored NP translations which it must operate on. In (15) there is nothing to prevent [accompany*)]$_0$ being retrieved first, despite the fact that it has no direct contri-

---

[2]. The observant *reader* will have noticed that the interpretation obtained in this way is synonymous with the *de re* reading already mentioned. This is since the NP *an agent of a company* is itself unambiguous. Ambiguous NP's (e.g. *an agent of every company)* would give rise to distinct readings.

bution to make to the partially complete sentential translation.

The obvious solution would be to amend the S-Retrieval rule so that it checks for the free occurrence of an address variable, prior to retrieving a given NP translation. An appropriate formulation is the following:

S-Retrieval (with free variable check):

> If α is an S node and the sequence $<\phi, \sigma^1,[NP']_i, \sigma^2>$ is an interpretation for α, *and* x_i *occurs unbound in* ϕ, then the sequence $<NP'({}^\wedge\lambda x_i.\phi), \sigma^1\sigma^2>$ is also an interpretation for α.

Aside from the fact that the new rule relies on a purely syntactic property of the logic used (i.e. whether or not an arbitrary expression of IL contains a given free variable), there is at least one place where it may still fail to produce sensible results. Such a case would be the sentence given in (16a) below, where the pronoun *him* is to be read co-referentially with *a boy*. The reason is that the usual translation given to a pronoun is an expression of the form $\lambda Q.Q\{x_i\}$, for some free variable x_i. To produce a logical translation for the sentence having the required interpretation this variable must be captured as a result of performing Storage Retrieval for the binding operator corresponding to *a boy*. A possible interpretation for (16a) which would achieve this is given in (16b), with *him* translated as $\lambda Q.Q\{x_0\}$. [3].

(16)

a. Every sister of a boy hates him.

b. $<hate'({}^\wedge\lambda Q.Q\{x_0\})(x_1),[every'(sister'({}^\wedge\lambda P.P\{x_0\}))]_1,[a'(boy')]_0>$

c. $Ex[boy'(x)\&\forall y[sister'({}^\wedge\lambda P.P\{x\})(y) => hate'({}^\wedge\lambda Q.Q\{x\})(y)]]$

d. $\forall x[sister'({}^\wedge\lambda P.P\{x_0\})(x) => Ey[boy'(y)\&hate'({}^\wedge\lambda Q.Q\{y\})(x)]]$

If *every sister (of him)* is retrieved first then everything works out fine, with the result as shown in (16c). This reading might be paraphrased: 'there is some particular boy and every sister of his hates him'. On the other hand, since x_0 now occurs free in $hate'({}^\wedge\lambda Q.Q\{x_0\})(x_1)$ on account of the translation of *him*, then the amended S-Retrieval

---

[3] Clearly this is not intended to be a particularly sophisticated treatment of pronouns, for one thing the translation of *him* given in the text makes no attempt to deal with matters of gender agreement. However, it does serve to illustrate the point.

rule can still apply first to *a boy* leading to the anomalous (I6d). A possible paraphrase for the latter would be: 'Every sister of him hates a boy*.

Fortunately there is an alternative approach to translation which neither relies on syntactic properties of the logic, nor falls foul of pronouns. The proposed solution is to introduce just enough structure into the store to make explicit the order in which binding operators may be retrieved. Similar suggestions have been made by Elizabet Engdahl [Engdahl 1980. 1986] in her work on the semantics of questions. To begin with a new NP-Storage rule is required:

NP-Storage (Nested Store):

> If c* is an NP node, and the sequence <*c*<\<*f*> is an interpretation for *o*<. then the sequence <XPJP{x^)X<^'»$^{(r}$>]*.> for some unique index **i** is also an interpretation for oe.

Rather than simply putting the denotation of an NP node into store, along with anything else that happens to be there, the revised rule creates a completely new store and places in it **the whole interpretation** for the NP: **<oc\<*f*>**. The intention is to record the fact that any binding operator in the sequence <r has a direct role to play in the expression *ex* , though not necessarily in any larger expression prior to retrieving <*x* itself. Clearly a new S-Retrieval rule is also required:

S-Retrieval (Nested Store):

> If oc is an S node and the sequence <^>,cr^{1}j[<p,<r>]$_{i}$,(r^{z}> is an interpretation for^c, then so is the sequence <f3CXx; •^),<5*^{1}<**cr^{1}>.

The modified S-Retrieval rule need make no reference to free variables. Note that whenever some item **p** is retrieved from store, then its associated sequence of binding operators *6** is 'added* to the items remaining in store at the S node. This **Storage Promotion,** to give it a name, **ensures** that anything stored during the translation of £ becomes accessible to the S-Retrieval rule once £ itself has been retrieved. Significantly. S-Retrieval **cannot** apply to the sequence of stored binding operators associated with *3* prior to Storage Promotion.

To see how this works in practice the troublesome example of section 3 — *John* *seeks an agent of a company* — will be re-analysed. It will be remembered that Cooper's strategy failed when both NP denotations were stored. This lead to the possibility of retrieving the denotation of *a company* first, resulting in a spurious interpretation of the sentence. Before describing how the new storage strategy overcomes this problem, it should first be checked that the three valid readings can still be obtained.

Firstly it is clear that without invoking NP-Storage the narrow-scope reading of the sentence may be obtained as before. Modulo some very minor changes of detail, the wide-scope interpretation is also produced as previously with a single application of NP-Storage to the whole object term. Likewise, the third reading (on which *a company* is read *de re)* will result just in case NP-Storage is only applied to the embedded NP.

Choosing to store both NP denotations, the sequence below is first constructed for *an agent of a company.*

(17) <atagentOPJ$^{>}${x$_0$}))J[<a$^{>}$(company')>]$_D$>

Here there is a single binding operator in store corresponding to the interpretation of *a company.* Storing (17) in turn results in (18).

(18) <\Q.Q{x$_t$*)l*<a^f(agent'CXP.P{x$_o$ })),[<accompany*)>]$_0$ >]$_t$>

The new sequence again has a single item in store derived from the previous interpretation (17). Translation now continues as before yielding the result:

(19) $\langle seek'({}^\wedge\lambda Q.Q\{x_1\})(j),[\langle a'(agent'({}^\wedge\lambda P.P\{x_o\})),[\langle a'(company')\rangle]_o\rangle]_1\rangle$

At this stage, the S-Retrieval rule can be applied, but note that there is no longer any choice as to how to proceed. In fact only the expression a*(agent'C\PJ$^{>}${x$_o$})) is initially accessible to the rule. Following the first application the result is as shown in (20a). with Storage Promotion ensuring that accompany) becomes available for retrieval. A further application and the store is empty as in (20b) signaling that translation is complete.

(20)

a. <atagentrAPJP{$x_{c>}$})X^{A}Ax_1^eek'rXQ.Q{x_1}Xj))X<$a^{>}$(compajiy$^{>}$)>]$_{D}$>
b. $\langle a'(company')({}^{\wedge}\lambda x_0 . a'(agent'({}^{\wedge}\lambda P.P\{x_0\}))({}^{\wedge}\lambda x_1 . seek'({}^{\wedge}\lambda Q.Q\{x_1\})(j)))\rangle$

The translation thus obtained is equivalent to (13) of section 3. The fact that S-Retrieval cannot proceed in the opposite order guarantees that the unwanted reading (12) cannot arise.

The example involving a pronoun can be treated in a like manner. The sentence is repeated below along with a suitable representation.

(21)
a. Every sister of a boy hates him.
b. $\langle hate'({}^{\wedge}\lambda Q.Q\{x_0\})(x_1), [\langle every'(sister'({}^{\wedge}\lambda P.P\{x_0\})), [\langle a'(boy')\rangle]_0\rangle]_1\rangle$

Once again there is no choice as to which stored NP denotation may be retrieved first. It is only possible to start with every*(sister*CAPJP{xe>})) and then retrieve a*(boy*) following Storage Promotion. In consequence both occurrences of x_o will become bound as required.

5. *Computational Issues and Summary.*

Computational linguists have already taken advantage of Cooper's storage technique for dealing with quantifier scope ambiguity in the design of natural language processing systems [Gavron *et al* 1982] [Keller 1984]. The strategy is particularly appealing because it relies on the principle of surface compositionality. enabling semantic representations to be built up in parallel with syntactic analysis. In the Hewlett-Packard system of Gavron *et al,* Cooper Storage is utilised in the generation of first order Logical Representations during parsing. These Logical Representations are then mapped into expression of the query language for HIRE, a relational database. This allows the further possibility of a principled approach to ambiguity resolution on the basis of 'state of the world* knowledge as. say. reflected in the database.

Nested Cooper Storage presents no particular difficulties with regard to implementation. Indeed, a substantially correct treatment of quantifier scope ambiguity which is more restrictive than competing approaches in that it predicts fewer readings is clearly to be favoured computationally. This point has been stressed by Hobbs and Shieber in a recent paper [Hobbs and Shieber 1986] which presents a new algorithm for generating quantifier scopings. Hobbs and Shieber specifically consider noun phrases introducing multiple quantifiers of the kind exemplified in section 3 of this paper. It is shown that in contrast to 'naive' algorithms which generate all possible permutations of quantifiers, the new algorithm represents a considerable saving in computational effort. For example, a naive approach to sentence (22) below, which introduces the five quantifiers *some*, *every*, *most*, *a* and *each* predicts that it has no fewer than 120 (i.e. 5 factorial) readings.

(22) Some representatives of every department in most companies saw a few samples of each product.

It should be clear from the discussion of Cooper Storage in section 2 of this paper that any faithful implementation of Cooper's strategy will also be naive in this sense. Modulo *de dicto/de re* ambiguities it will make precisely the same predictions. In fact, of these 120 readings, only those 42 generated by the algorithm of Hobbs and Shieber turn out to be valid. Again, in contrast to naive algorithms, a faithful implementation of Nested Cooper Storage will generally predict fewer readings. Interestingly, it seems that it will produce precisely those orderings of quantifiers generated by the Hobbs and Shieber algorithm.

In summary, it has been demonstrated that Cooper Storage fails to provide a satisfactory treatment of quantifier scope ambiguity in certain cases. More specifically, overgeneration can arise in the context of complex NP's. A proposed modification of Cooper's translation strategy, involving a nested store and Storage Promotion overcomes this problem.

Aside from some limited assumptions regarding constituent structure Nested Cooper Storage is not dependent upon a particular view of syntax nor even particular syntactic analyses. The result is a principled approach to quantifier scope ambiguity compatible with much contemporary research on syntax. As such it should be clear that Nested Cooper Storage makes no claims with regard to preferred readings. Yet a systematic account of how native speakers choose a particular interpretation must surely proceed from an empirically justified account of the available choices.

Finally it is noted that a substantially correct treatment of quantifier scope ambiguity which is more restrictive than previous approaches (in the sense that it predicts fewer readings) is clearly to be favoured computationally. For this reason it is expected that Nested Cooper Storage will be of interest to computational linguists.

References.

Chomsky, N. (1970). Remarks on nominalization. In Roderick A, Jacobs and Peter S. Rosenbaum. *eds.* Readings in English Transformational Grammar. Waltham, Massachusetts: Ginn and Company, pp. 184-221.

Cooper, R. (1975). Montague's semantic theory and transformational syntax. Ph.D dissertation, University of Massachusetts at Amherst.

Cooper, R. (1983). Quantification and Syntactic Theory. D. Reidel Dordrecht Holland.

Engdahl. E (1980). The Syntax and Semantics of Question in Swedish, unpublished Ph.D dissertation. University of Massachusetts Amherst.

Engdahl. E (1986). Constituent Questions: The Syntax and Semantics of Questions with Special Reference to Swedish. D. Reidel Dordrecht Holland.

Gawron. J. M., J. King, J. Lamping, E. Loebner, A. Paulson, G. Pullum, I. Sag & T. Wasow. (1982) Processing English with a Generalised Phrase Structure Grammar. Proceedings of the 20'th Annual Meeting of the Association for Computational Linguistics, University of Toronto, June.

Gazdar, G., E. Klein, G. Pullum and I. Sag (1985). Generalized Phrase Structure Grammar. Oxford: Basil Blackwell.

Hobbs, J. and S. Shieber (1986). An Algorithm for Generating Quantifier Scopings. SRI Research Paper.

Keller, W. R. (1984). Generating Logic from ProGram Parse Trees. Cognitive Studies Research Paper, Serial no. CSRP 39. University of Sussex.

Montague, R. (1974a). Universal Grammar. In Thomason, Richmond, ed. Formal Philosophy: Selected Papers of Richard Montague. New Haven: Yale University Press, 1974, pp.222-246.

Montague, R. (1974b). The Proper Treatment of Quantification in Ordinary English. In Thomason, Richmond, ed. Formal Philosophy: Selected Papers of Richard Montague. New Haven: Yale University Press, 1974, pp.247-270.