Finite-state methods, binding, and anaphora

Dick Oehrle[1]
Linguistics & Cognitive Science
University of Arizona
rto@chol.douglass.arizona.edu

Broadly construed, dynamic semantics views interpretation as a process that is both context-dependent and context-affecting. Context-dependence is illustrated most dramatically by indexicals; context-affectingness by the interpretation of indefinites. But the framework is quite general: schematically, we can represent the interpretation of an expression α as

$$^{C}[\,\alpha\,]^{C'}$$

where 'C' represents the input context and 'C'' the output context. This schematic representation raises obvious questions:

1. what are the relevant aspects of context? That is, how should C and C' be structured?
2. in what ways can the interpretation of α depend on C?
3. what transitions between C and C' are possible?
4. how do these transitions depend on α?
5. given a particular occurrence of α, how is the context C on which it depends determined?

Answering these questions in different ways fills in a broad landscape of different approaches to binding and anaphora: for example, we may take the contexts C and C' to be Discourse Representation Structures [8], variable assignment functions [6], or a pair of sets representing the global c-commmanders and local c-commanders that the GB Binding Theory [3] depends on.

In this paper, we sketch how a dynamic account of English binding and anaphora can be formulated, illustrating how structured contexts can be constructed and accessed as a by-product of grammatical composition in a system of labeled deduction [12]. Our focus is on the applicability of finite-state methods to the dynamics of interpretation in this system, motivated by several considerations. By the general schema above, interpretation effects a transition between two contexts — a property with clear affinities to the transitions of automata. An obvious question, then, is what the properties of these transitions are viewed from the perspective of automata. At the same time, investigating such questions leads to accounts of natural language binding phenomena which are constructive in nature, and thus of independent interest.

[1] This material is based upon work supported by the National Science Foundation under Grant No. SBR-9510706. I'd like to express my gratitude to András Kornai, Michael Moortgat, and Susan Steele for advice and encouragement.

© 1996 R. Oehrle
Proceedings of the ECAI 96 Workshop
Extended Finite State Models of Language
Edited by A. Kornai.

1 Background

Consider a sentential form such as the following:[2]

$$np_i \textit{ told } np_j \textit{ that } np_k \textit{ called } np_l$$

This sentential form offers a small laboratory in which we may investigate how to characterize referential relations in English.

As a first step, we indicate the way the GB Binding categories bear on the interpretation of the referential positions in this sentential form. We depart from the standard GB Binding Theory in allowing anaphors in the subject position of the embedded clause — indicated in the fifth column of the anaphor row — because reciprocals seem to occur in this position (bound by elements in the matrix clause) although reflexive forms do not. Following Brame [2], we regard this difference as arising from the fact that reflexives have a fixed case incompatible with the contexts that accommodate nominative or possessive pronouns. We write $i \mapsto j$ to indicate that any admissible assignment of values to indices in a model structure, the index i is assigned to the value that j is assigned, and $i \not\mapsto j$ to indicate that admissible assignments assign distinct values to the indices i and j; if A is a set of indices, $i \mapsto j \in A$ means that any admissible assignment of values to indices is consistent with $i \mapsto j$, for some $j \in A$, and $i \not\mapsto A$ means $i \not\mapsto a$, for all $a \in A$.

	np_i	np_j	np_k	np_l
anaphor:	*	$j \mapsto i$	$k \mapsto \{i,j\}$	$l \mapsto k$
pronominal:	[free]	$j \not\mapsto i$	[free]	$l \not\mapsto k$
R-expr.:	[free]	$j \not\mapsto i$	$k \not\mapsto \{i,j\}$	$l \not\mapsto \{i,j,k\}$

Each cell of the table indicates how the interpretation of an occurrence of a particular kind of *np* in one of these *np*-positions is constrained. For example, take the instantiation of this sentential form below, on the assumption that *Smith* and *Jones* are R-expressions and *he* and *her* are pronominals:

$$\textit{Smith}_i \textit{ told Jones}_j \textit{ that he}_k \textit{ called her}_l.$$

[2] Here and below, we use indices as *referential indeterminates:* each *occurrence* of a referential expression whose referent is not assumed to be a constant in any discourse is assigned a *fresh index*. Binding of anaphoric expressions is not dependent on identity of indices, but rather by *anchoring* an index to a constant or to another index.

According to the above table, we may represent the constraints on the interpretation of this sentence as:

$$j \not\mapsto i \wedge l \not\mapsto k$$

This constraint is consistent with an interpretation on which $k \mapsto i$ and $l \mapsto j$ and with an interpretation on which $k \mapsto j$ and $l \mapsto i$, as well as with interpretations on which one or both of k, l are independent of both i and j.

In the Binding Theory of the GB tradition, the set of admissible anchors for any occurrence of an anaphoric expression is regulated in a way directly dependent on global properties of postulated syntactic structures [3]. It is common practice in this tradition to view anaphoric anchoring as analogous to binding a free variable by a quantifier in first-order logic or by the abstraction operator of λ-calculus. In the binding theory of [3], two properties of the context in which an anaphoric expression occurs play a role in its interpretation:

- local c-commanders
- all c-commanders

The local c-commanders occur in the constraints associated with anaphors and pronominals (Principles A and B of [3, p. 20] below), and the set of all c-commanders plays a role in Principle C, since to assert that an expression is free in an absolute sense is equivalent to asserting that it is not anchored to any c-commander, local or not. Chomsky [3] states these rules in the following, familiar form:

DEFINITION. An expression is *bound* in a domain D if it is co-indexed by a c-commanding category occurring in D; otherwise, it is *free* in D.

BINDING PRINCIPLES.

A. An *anaphor* is bound in its governing category.
B. A *pronominal* is free in its governing category.
C. An *R-expression* is free.

anaphors:	*himself, herself, ..., each other*
pronominals:	*her, him, ..., he (?), his(?), ...*
pronominal anaphor:	PRO
R-expressions:	*John, Dan, Stan, ...*

1.1 A dynamic view of GB binding

We now recast this account informally from the point of view of dynamic semantics. There are four essential steps:

- an appropriate notion of input and output context must be chosen;
- each lexical expression must be associated with a dynamic interpretation (a relation between input contexts C, evaluations relative to C, and output contexts C');
- each occurrence of a lexical expression must be associated with a specific input context;
- a method of determining the complete set of interpretations of an expression must be stated.

Our goal will be to show how these steps can be realized in principle in a general way in a system of labeled deduction in which English expressions are associated with functions from contexts to finite-state machines which recognize a term of Montague's intensional logic *IL* [15] for each of its interpretations. Within this general system, a variety of different accounts of quantification and binding can be realized. Since the GB system is widely familiar, however, it provides an appropriate introduction to this program.

It is a common practice to label *np*-positions with an *index*, indicating its referential potential. Moreover, it is possible as well to label each index with a further pair of indices, indicating the set of indices associated with globally c-commanding *np*'s and locally c-commanding *np*'s. For a given index i, global c-commanders G and local c-commanders L, we will write the resulting labeled index:

$$\overset{G}{L}\,[i]$$

Accordingly, an *np* labeled with a context-labeled index can be represented in the format *type: labeled index*:

$$np : \overset{G}{L}\,[i]$$

Now, choosing fresh variables for each *np*-position, and annotating each *np*-position with morphological information, we can represent the contextual contribution of our laboratory frame as follows:

$$np[nom] : \overset{\emptyset}{\emptyset}\,[x] \quad \textsf{told} \quad np[acc] : \overset{\{x\}}{\{x\}}\,[y] \quad \textsf{that}$$
$$np[nom] : \overset{\{x,y\}}{\{x,y\}}\,[z] \quad \textsf{called} \quad np[acc] : \overset{\{x,y,z\}}{\{z\}}\,[w]$$

Note that the initial *np* is labeled by an index whose global c-commanders and local c-commanders are both empty. The object-position of *told* is associated with an index whose set of global c-commanders and set of local c-commanders are both the singleton set consisting of the index of the subject position of *told*. The labels of the embedded subject reflect all the indices of the higher clause. But the object-position of the embedded clause reflects a change: the local c-commanders of its index consist of the singleton set containing the index of its clausemate subject. This context-labeling conforms with the empirical assumptions of the GB Binding Theory presented earlier.

It is also possible to label various expressions in such a way as to characterize their compatibility with particular positions in this laboratory example. The lexical assumptions presented below take the form

$$\textsf{orthographic\_term} : \quad type : \quad \overset{G}{L}\,[index]$$

Using '_' as an anonymous local variable over the grammatical cases, we start with names, reflexive pronouns, and nominative and accusative non-reflexive pronouns. For present purposes, we may regard the index associated with proper names as a referential constant. Otherwise, the variables should be regarded as analogous to Prolog program variables: they will unify with context-labels where possible, and be instantiated to fresh variables otherwise. The symbol $A \setminus B$ denotes the relative complement of B in A.

names	smith:*np[_]*: ${}^{A\setminus\{s\}}_{A'\setminus\{s\}}[s]$
	jones:*np[_]*: ${}^{A\setminus\{j\}}_{A'\setminus\{j\}}[j]$
	⋮
reflexives	herself:*np[acc]*: ${}^{A'}_{A\cup\{i\}}[i]$
	⋮
acc. pronouns	her:*np[acc]*: ${}^{A'}_{A\setminus\{i\}}[i]$
	⋮
nom. pronouns	she:*np[nom]*: ${}^{A\cup A'}_{A}[i]$

The schema associated with the name smith, for example, assigns it the interpretation s (regarded as a constant of type e in Montague's IL) in contexts in which s belongs to neither the set of local c-commanders nor the set of global c-commanders. A reflexive form such as herself can be assigned interpretation i only when i belongs to the set of local c-commanders. Accusative pronouns can be assigned interpretation i only when i does not belong to the set of local c-commanders. And nominative pronouns are unconstrained.

Now, consider on these assumptions the possibility of various referential relations for sentences resulting from the substitution of these lexical elements into the different positions of our laboratory frame. For example, the sentence Smith$_s$ told Jones$_j$ that he$_k$ called him$_l$ has an interpretation in which k is bound to s and l is bound to j, because there is a global unifier unifying the labeled indices of the frame positions with the corresponding labeled indices of the lexical assumption associated with them. In the two blocks of the table below, the first line represents the constraints imposed by the frame; the second represents the constraints imposed by the lexical assumption; and the third line represents a unifier for the first and second lines:

$np[nom] : {}^{\emptyset}_{\emptyset}[x]$ *told*	$np[acc] : {}^{\{x\}}_{\{x\}}[y]$ *that*
smith : ${}^{A\setminus\{s\}}_{A'\setminus\{s\}}[s]$	jones : ${}^{B\setminus\{j\}}_{B'\setminus\{j\}}[j]$
${}^{\emptyset}_{\emptyset}[s]$	${}^{\{s\}}_{\{s\}}[j]$
$np[nom] : {}^{\{x,y\}}_{\{x,y\}}[z]$ *called*	$np[acc] : {}^{\{x,y,z\}}_{\{z\}}[w]$
he : ${}^{C}_{C'}[i]$	him : ${}^{D}_{D'\setminus\{k\}}[k]$
${}^{\{j,s\}}_{\{j,s\}}[s]$	${}^{\{j,s\}}_{s}[j]$

There are other unifiers as well, corresponding to other interpretations for the pronouns he and him. But the interpretations of pronouns and names in this simple laboratory example satisfies the principles of the Binding Theory presented earlier, even though the mechanisms employed are not the same as those assumed in that tradition.[3]

It remains to be seen how the contextual labeling of the frame itself is built up. In a categorial setting, this task can be accomplished by assigning suitable structure to functor types. In an earlier paper [12], for example, the types for called, that, and told were chosen as follows:

$$\lambda\, \mathsf{x}\ \lambda\, \mathsf{y}\ .\ \mathsf{x\ called\ y}: \quad np[nom] \overset{\otimes}{\to} np[acc] \overset{\otimes}{\to} s :$$
$$\lambda\ {}^{G}_{L}[x] \lambda\ {}^{G\cup\{x\}}_{\{x\}}[y]\ {}^{G}_{L}[called(x,y)]$$

$$\lambda\, \mathsf{z.\ that\ z}: \quad s \overset{\otimes}{\to} s' : \lambda\ {}^{G}_{L}[z].\ {}^{G}_{L}[z]$$

$$\lambda\, \mathsf{x}\ \lambda\, \mathsf{y}\ \lambda\, \mathsf{w.\ x\ told\ y\ w}: \quad np[nom] \overset{\otimes}{\to} np[acc] \overset{\otimes}{\to} s' \overset{\otimes}{\to} s$$
$$\lambda\ {}^{G}_{L}[x] \lambda\ {}^{G\cup\{x\}}_{\{x\}}[y] \lambda\ {}^{G\cup\{x,y\}}_{\{x,y\}}[w]\ {}^{G}_{L}[told(x,y,w)]$$

These functors act simultaneously in three dimensions: strings, types, terms. The type for called, for example, combines with two np's, each labeled with string information and an interpretive term, to form a sentence s. The strings associated with the two np's combine through application and normalization to the left and the right of the string called to form the string associated with s; the interpretation associated with s also results from application and normalization in a way that requires unification of context labeling. In this way, each referential expression is evaluated relative to an appropriate context. Details can be found in [12].

In what follows, we follow the same strategy: functor expressions provide the vehicle to construct context; argument expressions are interpreted in a way sensitive to context. But the choices involved in both construction and interpretation are implemented in finite-state machines.

2 Labeled deduction

The Curry-Howard correspondence [5, 4] assigns to each proof in (a subsystem of) intuitionistic logic a term of (a subsystem of) the λ-calculus. For natural deduction proofs, the correspondence is a bijection in which removal of detours corresponds to β-reduction. For the Gentzen-style sequent presentation, the correspondence is many-to-one. For occurrence logics, the equivalence relation among sequent proofs determined by this correspondence has another characterization: two proofs of an endsequent Σ correspond to the same λ-term (up to alphabetic variance) iff their axiom leaves bind the same pairs of atomic sub-formulas of Σ. (For discussion, see [11].) What this means for our purposes is that the Curry-Howard correspondence distinguishes differences in application and pairing, but does not distinguish finer-scale distinctions in referential relations.

To represent these finer-scale distinctions, we will assign to each proof a function from input contexts to a family of IL-terms. Which terms? For each proof and a fixed context, these terms will be the language of a finite machine. Moreover, we will be able to construct the finite machine associated

[3] In particular, we follow Brame [2] and Pollard & Sag [14] in assigning reflexives intrinsic accusative case which blocks them from appearing in nominative or genitive positions. Accordingly, for English, we can allow the nominative pronouns (and the genitive pronouns) to impose no restrictions on their context: they can be bound by c-commanders or free.

with a proof so that it imposes appropriate restrictions on the interpretation of pronouns and anaphors.

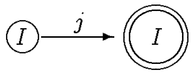
To carry out this program, we start by examining some simple finite state machines. This will allow us to define the interpretation of atomic referential expressions as functions from contexts to a pair consisting of an interpretation and an output context.

2.1 Individuals and machines

Let $I = C \sqcup P$ be a finite set of *individuals*, partitioned in *constants* C and *parameters* P. There are some simple machines that it will be useful to define. The vocabulary Σ over which these machines are defined is a set in 1-1 correspondence with I.

2.1.1 Characteristic machines

Each subset J of I determines a finite-state machine $\mathcal{M}_J$, called a *characteristic machine*. Its states consist of the initial state I and for each $j \in J$ an arc labeled j from the initial state I to a unique final state also labeled I, as indicated below:



These machines recognize exactly the elements of J and fail on the relative complement of J in I. Moreover, characteristic machines inherit Boolean operations from the power set $Pow(I)$ in the obvious way:

$$\begin{aligned}\mathcal{M}_J \cup \mathcal{M}_K &= \mathcal{M}_{J\cup K}\\ \mathcal{M}_J \cap \mathcal{M}_K &= \mathcal{M}_{J\cap K}\\ \mathcal{M}_J - \mathcal{M}_K &= \mathcal{M}_{J-K}\end{aligned}$$

2.1.2 Constant machines

Now for any constant $c \in I$, and any subset J of I, there is a *constant machine* $\mathcal{M}^c_J$ whose initial state is J and which contains exactly one arc, labeled c, from the initial state to the final state $J \cup \{c\}$. Constant machines succeed on the constant c, for any subset J of I, and fail on any other constant or parameter.

2.1.3 Parameters

Finally, we need machines which will succeed when a fresh parameter is added to a set of individuals. For this purpose, it is useful to suppose that the finite set of parameters P contained in the set of individuals I is drawn from a set of parameter symbols indexed by the natural numbers. Then, given a subset J of I, the *parameter machine* $\mathcal{M}^P_J$ can be defined as the machine whose initial state is J and which contains exactly one arc, labeled with that parameter $p \notin J$ with minimal index, from the initial state to the final state $J \cup \{p\}$. (Note that if every parameter of I belongs to J, we must augment our background set of individuals to $I \cup \{p\}$, which is still finite.)

2.1.4 Observations

These machines are all extremely simple. Ignoring failures, there is a single initial state; all successful paths traverse a single arc; final states are in a 1-1 correspondence with successful arcs; thus, they simply consist of finite trees of depth 1, with root the initial state and leaves the final states.

2.2 Contexts

Abstractly, we take a *context* to be an n-tuple of subsets of I. If a subset of individuals J is one of the members of the context $\mathcal{C} = C_i(1 \leq i \leq n)$, the definitions above have an obvious extension. For example, we write

$$\mathcal{C} \xrightarrow{\mathcal{M}_{\mathcal{J}}} \mathcal{C}$$

for the machine whose initial state is labeled with $\mathcal{C}$, and each of whose arcs is an edge labeled with a distinct element $d \in D$ from the initial state to a final state labeled $d;\mathcal{C}$. Thus, the set of edges and set of states corresponds precisely to the edges and states of $\mathcal{M}_D$; the only difference is that in the labels of the states, D is replaced everywhere by $\mathcal{C}$.

Similarly, for constant machines and parameter machines, with minor changes. In the case where J is an element of $\mathcal{C}$, we denote the intended extension of the constant machine $\mathcal{M}^c_J$ by

$$\mathcal{C} \xrightarrow{\mathcal{M}^c_J} \mathcal{C}'$$

where $\mathcal{C}'$ differs at most from $\mathcal{C}$ by containing $J \cup \{c\}$ in place of J. In the same way, we denote the intended extension of the parameter machine $\mathcal{M}^P_J$ by

$$\mathcal{C} \xrightarrow{\mathcal{M}^P_J} \mathcal{C}'$$

where $\mathcal{C}'$ differs from $\mathcal{C}$ by containing $J \cup \{p\}$ in place of J

This notation provides a way of denoting functions from contexts to machines: specifying the context $\mathcal{C}$ fixes a particular machine. In fact, such functions provide a convenient way to assign interpretations to particular referential expressions.

2.2.1 Linguistic discussion

As a first approximation, assume that the relevant notion of context is a triple $\langle L, G, D\rangle$, where D is a set of accessible individual discourse referents for an occurrence of an expression, with L the set of referents of local c-commanders of the expression and G the set of referents of its global c-commanders.

For a proper name such as Smith, statically interpreted as denoting individual s, these three parameters determine the machine:

$$\langle L, G, D\rangle \xrightarrow{\mathcal{M}^s_D - \mathcal{M}_{G\cup L}} \langle L, G, D \cup \{s\}\rangle$$

For a reflexive such as herself, these three parameters determine the machine:

$$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_L} \langle L, G, D\rangle$$

For a definite accusative pronoun such as her, these three parameters determine the machine:

$$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_D - \mathcal{M}_L} \langle L, G, D\rangle$$

For a nominative or possessive pronoun such as he or his, these three parameters determine the machine:

$$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_D} \langle L, G, D\rangle$$

Note here that our notion of context is slightly richer than the notions explicitly referred to in Principles A, B, and C of the GB account. But the additional parameter D is necessary to constructively specify the interpretation of non-reflexive pronouns. Later, when we consider quantifiers, we will see that it is useful to regard D as the union of two disjoint sets: a set D_c of accessible constants and a set D_p of accessible parameters.

Another class of expressions we shall need involves non-referential constants: consider any non-referential expression ϕ, statically interpreted as an IL constant ϕ'. For any two context $\mathcal{C}$, there is a machine $\mathcal{M}_{\phi'}$ with initial state labeled $\mathcal{C}$, final state labeled $\mathcal{C}$ and a single edge from $\mathcal{C}$ to $\mathcal{C}$ labeled ϕ'.

It is worth noting that these machines are defined for any input context.

Finally, observe that the labels on the output final states in all of these cases are completely determined by the input context and the class of machines involved. As a result, it is not necessary to indicate the output context and we can denote these cases more succinctly as follows:

names:	$\langle L, G, D\rangle \xrightarrow{\mathcal{M}^s_D - \mathcal{M}_{G \cup L}}$
reflexives:	$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_L}$
acc. pronouns:	$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_D - \mathcal{M}_L}$
nom. pronouns:	$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_D}$
non-ref. constants:	$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_{\phi'}}$

2.3 Composition of machines

Any two finite-state machines can be concatenated to form a new finite-state machine. If we regard the interpretation of expressions as functions from contexts to machines, then two such functions should compose in a way that depends on their concatenation. Indeed, since all of the functions that we need are defined for all contexts we shall consider, and since every final state is labeled with a contextual specification, given two machines

$$\mathcal{C} \xrightarrow{\mathcal{M}_1} \quad \text{and} \quad \mathcal{D} \xrightarrow{\mathcal{M}_2}$$

it is possible to attach a copy of $\mathcal{M}_2$ to every final node of $\mathcal{M}_1$, taking the input context of $\mathcal{M}_2$ to be the context specified by the node to which it is attached. We will denote the result of this operation by

$$\mathcal{C} \xrightarrow{\mathcal{M}_1 ; \mathcal{M}_2}$$

It is also possible to concatenate two functions from contexts to machines in such a way that the second machine is relativized to a different context than the context specified at any final node of the first to which it is attached. If $f : \mathcal{C} \to \mathcal{C}'$ is the specification of such a context — a function from contexts to contexts — we write

$$\mathcal{C} \xrightarrow{\mathcal{M}_1 ;_f \mathcal{M}_2}$$

for the function from contexts to machines which results from running $\mathcal{M}_1$ on its argument and then attaching to each final node with context label D the machine that results from running $\mathcal{M}_2$ on $f(D)$.

To motivate this definition in part, consider a transitive verb like blames. This combines with a nominative np to its left and an accusative np to its right for form an s. We can specify this much by assigning blames the category

$$np[nom]\backslash(s/np[acc])$$

But this fails to represent the various referential dependencies that anaphoric interpretation requires. In particular, let us assume that these dependencies satisfy the following conditions:

- the local c-commanders of the input context to $\mathcal{M}_{np[nom]}$ = the local c-commanders of the input context to $\mathcal{M}_s$;
- the global c-commanders of the input context to $\mathcal{M}_{np[nom]}$ = the global c-commanders of the input context to $\mathcal{M}_s$;
- the discourse referents of the input context to $\mathcal{M}_{np[nom]}$ = the discourse referents of the input context to $\mathcal{M}_s$;
- the local c-commander of the input context to $\mathcal{M}_{np[acc]}$ = the referent of $np[nom]$; of the input context to $\mathcal{M}_s$;
- the global c-commanders of the input context to $\mathcal{M}_{np[acc]}$ = the union of the referent of $np[nom]$ and the global c-commanders of the input context to $\mathcal{M}_s$
- the discourse referents of the input context to $\mathcal{M}_{np[acc]}$ = the union of the discourse referents of the input context to $\mathcal{M}_{nom}$ and the discourse referents of the output context to $\mathcal{M}_{np[nom]}$.

We can specify these dependencies in the category for blames by annotating its sub-types with appropriately connected input and output contexts. For type A with input schema κ, we write

$$^{\kappa}A$$

By convention, we can refer to the output state of any final node of a machine associated with $^{\kappa}A$ as κ'.

In the case at hand, we take the machine associated with the verb blames to be the constant machine

$$\mathcal{C} \xrightarrow{\mathcal{M}_{blames}}$$

which for any context succeeds on the constant $blame$ of type $\langle e, \langle e, t\rangle\rangle$ and returns the same context.

Suppose that we combine this verb with the subject first and that the subject is associated with the machine

$$\mathcal{C} \xrightarrow{\mathcal{M}_1}$$

We will associate the subject verb combination with the machine:

$$\mathcal{C} \xrightarrow{\mathcal{M}_{blames};\mathcal{M}_1}$$

Note that the input context passed to $\mathcal{M}_1$ will be determined by the context argument represented by $\mathcal{C}$.

Now, if the machine associated with the object is

$$\mathcal{C} \xrightarrow{\mathcal{M}_2}$$

then we can associate the sentence resulting from combining blames with subject and object np's by

$$\mathcal{C} \xrightarrow{\mathcal{M}_{blame};\mathcal{M}_1;\langle i,\cup i,\cup i\rangle}$$

where '$\langle i, \cup i, \cup i\rangle$' represents the function from contexts to contexts which sets the input context for $\mathcal{M}_2$ at any final node of $\mathcal{M}_1$ to be the triple $\langle i, G' \cup i, D' \cup i\rangle$, with i the label of the $\mathcal{M}_1$ path reaching this final node in $\mathcal{M}_1$, and G' and D' to the output labels at this node for the second and third context coordinates.

We can specify this information categorially by annotating the type for blames as follows:

$$^{\langle L,G,D\rangle}np[nom]_i\backslash(s/^{\langle i,G'\cup i,D'\cup i\rangle}np[acc]_j)$$

Here the superscripts represent input contexts and the subscripts the various paths to successful outputs. All the other contextual information is determined by these parameters.

2.4 Examples

As an example, consider the following lexicon, whose entries consist of triples of the form $\langle$string : *type* : $\mathcal{M}\rangle$, displayed vertically:

Smith: $np[-]$: $\langle L, G, D\rangle \xrightarrow{\mathcal{M}^s_D - \mathcal{M}_{G\cup L}}$

Jones:$np[-]$: $\langle L, G, D\rangle \xrightarrow{\mathcal{M}^j_D - \mathcal{M}_{G\cup L}}$

herself:$np[acc]$: $\langle L, G, D\rangle \xrightarrow{\mathcal{M}_L}$

her:$np[acc]$: $\langle L, G, D\rangle \xrightarrow{\mathcal{M}_D - \mathcal{M}_L}$

she:$np[nom]$: $\langle L, G, D\rangle \xrightarrow{\mathcal{M}_D}$

blames:$^{\langle L,G,D\rangle}np[nom]_i\backslash(s/^{\langle i,G'\cup i,D'\cup i\rangle}np[acc]_j)$:
$\mathcal{C} \xrightarrow{\mathcal{M}_{blames}}$

called:$^{\langle L,G,D\rangle}np[nom]_i\backslash(s/^{\langle i,G'\cup i,D'\cup i\rangle}np[acc]_j)$:
$\mathcal{C} \xrightarrow{\mathcal{M}_{called}}$

told:$^{\langle L,G,D\rangle}np[nom]_i\backslash$
$\backslash((s/^{\langle i\cup j,G''\cup i\cup j,D''\cup i\cup j\rangle}s)^{\langle i,G'\cup i,D'\cup j\rangle}np[acc]_j)$:
$\mathcal{C} \xrightarrow{\mathcal{M}_{told}}$

Now, for a sentence of the form

np told np np called np

we have a sequence of types:

$np[nom]_x$
$^{\langle L,G,D\rangle}np[nom]_i\backslash((s/^{\langle i\cup j,G''\cup i\cup j,D''\cup i\cup j\rangle}s)^{\langle i,G'\cup i,D'\cup i\rangle}np[acc]_j)$
$np[acc]_y$
$np[nom]_u$
$^{\langle L,G,D\rangle}np[nom]_i\backslash(s/^{\langle i,G'\cup i,D'\cup i\rangle}np[acc]_j)$
$np[acc]_v$

On the assumption that implicational inferences (slash-eliminations) in the type system are accompanied by composition of machines, writing $\mathcal{M}_\alpha$ for the machine associated with the np indexed by α, we have as well the composition of machines indicated in Figure 1.

To consider the possibilities of intra-sentential anaphora for the main clause Smith told Jones she called her, we set all three argument parameters L, G, D to $\emptyset$, and make appropriate substitutions. The composition begins by running $\mathcal{M}_{told}$ on the input context:

$$\langle\emptyset,\emptyset,\emptyset\rangle \xrightarrow{\mathcal{M}_{told}}$$

This machine has a single arc labeled *told* and returns the empty input context, which forms the input to the machine for Smith, as depicted below:

$$\langle\emptyset,\emptyset,\emptyset\rangle \xrightarrow{\mathcal{M}^s_D - \mathcal{M}_{G\cup L}}$$

This machine also has a single arc labeled s and returns the context $\langle\emptyset,\emptyset,s\rangle$. But the input to the next step is not this context, but the context that results from replacing its first two coordinates with the path label on which the last machine succeeds, yielding:

$$\langle s,s,s\rangle \xrightarrow{\mathcal{M}^j_D - \mathcal{M}_{G\cup L}}$$

This succeeds on the single arc labeled j and returns the context $\langle s, s, s\cup j\rangle$. But the input to the next step increments the first two coordinates to $\langle s\cup j, s\cup j, s\cup j\rangle$ for the next step:

$$\langle s\cup j, s\cup j, s\cup j\rangle \xrightarrow{\mathcal{M}_{called}}$$

This succeeds on the single arc labeled *called* and returns the input context. In the next step, the input determines the machine for the nominative pronoun she as follows (taking $D = s\cup j$):

$$\langle s\cup j, s\cup j, s\cup j\rangle \xrightarrow{\mathcal{M}_{s\cup j}}$$

This has two successful arcs, one labeled s and one labeled j, returning the input context in each case. But the input to the next and last step is set to $\langle s, s, s\cup j\rangle$ and $\langle j, j, s\cup j\rangle$ in the second. Each of these determines the input to the machine for her (taking $D = s\cup j$ and $L = s$ in the first case, and $D = s\cup j$ and $L = j$ in the second), as indicated in the two steps below:

$$\langle s,s,s\cup j\rangle \xrightarrow{\mathcal{M}_{s\cup j} - \mathcal{M}_s}$$

$$\langle j,j,s\cup j\rangle \xrightarrow{\mathcal{M}_{s\cup j} - \mathcal{M}_j}$$

Thus, the sentence Smith told Jones she called her has two possible readings, corresponding to the two successful paths

$$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_{told} \,;\, \mathcal{M}_x \,;_{\langle x, G', D'\rangle}\, \mathcal{M}_y \,;_{\langle x\cup y, G'', D''\rangle}\, \mathcal{M}_{called} \,;\, \mathcal{M}_u \,;_{\langle u, G''', D'''\rangle}\, \mathcal{M}_v}$$

Figure 1. Composition of machines for np$_x$ told np$_y$ np$_u$ called np$_v$

through the composition of machines associated with its lexical components, namely:

told s j called s j
told s j called j s

If we substitute herself for her, the only difference is that at the last step the two final input contexts $\langle s, s, s \cup j\rangle$ and $\langle j, j, s \cup j\rangle$ are fed to the function from contexts to machines for reflexives:

$$\langle L, G, D\rangle \xrightarrow{\mathcal{M}_L}$$

In the first case, we have $L = s$; in the second, $L = j$. So the final steps are those shown below:

$$\langle s, s, s \cup j\rangle \xrightarrow{\mathcal{M}_s}$$
$$\langle j, j, s \cup j\rangle \xrightarrow{\mathcal{M}_j}$$

In this case, the resulting composite machine recognizes the two terms below:

told s j called s s
told s j called j j

2.5 Quantification

On the analysis suggested here, definite pronouns and reflexives do not introduce new discourse referents — they simply access existing discourse referents in a context-constrained fashion. Proper names are more flexible: the constants they are associated with may be either contained in the set of accessible discourse referents or they may be new to the discourse (a distinction correlated in part with accentual focus). Quantifier expressions have a different character. First, we will assume that each quantifier introduces a fresh parameter into the discourse. Second, the dynamic properties of quantifiers studied in DRT [8] and Dynamic Montague Grammar [6, 7] depend on the persistence of the parameter introduced. In the general framework introduced here, these properties can be straightforwardly modeled.

We take as fundamental the generalized quantifier perspective according to which a determiner such as every is to be interpreted by an *IL*-term of type $\langle\langle e, t\rangle, \langle\langle e, t\rangle, t\rangle\rangle$. There are a number of ways in which this basic assumption can be integrated into a system of categorial deduction. Here we follow the multi-modal analysis proposed by Moortgat [10], according to which a determiner is assigned the type

$$(\Diamond(s/_w(\Box^{\downarrow}np\backslash_w s)))/n$$

In this framework, the implicational types $/_w$ and $\backslash_w$ are the residuals of a binary 'wrapping' product $\circ$, whose interaction with the normal binary product $\bullet$ (of the non-associative Lambek calculus **NL**, say) allows the quantifer to find an appropriate scope with respect to a sentence missing an np in the position where the quantifier occurs. This behavior is mediated by the occurrence of the unary modal product $\Diamond$ and its residual $\Box^{\downarrow}$. For the details of this treatment, see Moortgat's paper [10].

According to the theory of generalized quantifiers, a determiner is a relation between two predicates, which we may indicate as follows:

$$Qx(Q, P)$$

What the dynamic properties are to be associated with such a schema? We mentioned earlier the usefulness of dividing the set of discourse referents into two disjoint sets — the set of discourse constants D_c and the set of discourse parameters D_p.[4] We will suppose, then, that our contexts have four coordinates $\langle L, G, C, P\rangle$:

- local c-commanders L;
- global c-commanders G;
- discourse constants D_c (or simply C); and
- discourse parameters D_p (or simply P).

Dropping the modalities from the formula $(\Diamond(s/_w(\Box^{\downarrow}np\backslash_w s)))/n$, since they are semantically neutral, we have $(s/_w(np\backslash_w s))/n$. Each of the subformulas of this part is associated with a subformula of the *IL* term representing the interpretation of a quantified sentence, as displayed below:

$(s_1/_w(np\backslash_w s_2))/n$	Q
n	Q
$s_1/_w(np\backslash_w s_2)$	$Qx(Q, -)$
$np\backslash_w s_2$	P
np	x
s_2	$P(x)$
s_1	$Qx(Q, P)$

What this means is that just as in the case of other functor categories, we can control the relativization of these various pieces of interpretation by annotating the subformulas of the determiner type. We sketch how this might be done.

First of all, the parameter x must be fresh with respect to the discourse context occurrence associated with the position of the np. This in fact is a consequence of the rule of hypothetical reasoning governing the introduction of the parameter. Since any element of L or G is always an element of either C or P, a consequence of this fact is that a quantifier in object position can never be construed as binding a pronoun in subject position even though it can take wide scope over it: this is the 'strong cross-over' violation observed in sentences

[4] This distinction is motivated by the contrast between such cases as the following, observed by researchers in DRT:
Every student who meets Jones admires her. She's a great professor.
Every student who meets a professor admires her. She's an inspiration.
In the first case, the name Jones introduces a constant that can anchor the occurrence of the pronouns her, in the same sentence, and she, in the following sentence. But the parameter introduced by a professor can serve as anchor only for her, not for she.

such as He blames every striker, where the reading on which he is bound by every striker is unavailable.[5]

Second, the parameter x must belong to the set of local c-commanders which forms the input context to the interpretation of n. This is motivated by the interpretation of such expressions as Smith admires every admirer of himself, in which the reflexive in admirer of himself is preferentially anchored to the parameter introduced by the quantifier. Examples such as He called every person Smith met, where the referent of he cannot be bound to the referent of Smith, regardless of the scope of the quantifier, suggest that the second coordinate of the input to n be identified with the second coordinate of the input of the parameter-introducing np. The set of discourse referents accessible in the input to n is a more delicate question, which we postpone to another occasion.

Third, note that although the parameter introduced must be fresh with respect to local c-commanders of the np argument, other components of the predicate representing the scope of the quantifier can be anchored to it. Thus, in a sentence such as A picture of its author graces each book's cover, the pronoun its can be anchored to the parameter introduced by the quantifier each book, on the condition that the quantifier takes wide scope with respect to the subject.

Fourth, a question widely studied in dynamic semantics is whether the parameter introduced by a quantifier is discourse persistent. This question is intimately related to the dynamic interpretation of logical connectives associated with conjunction, disjunction, implication, and negation. From the present perspective, what is important is that this difference in behavior can be modeled by passing or not passing along to the output context the parameter introduced by the quantifier (and any other new parameters introduced in the output context of the scope of the quantifier, represented by the output coordinate P). On the other hand, we will assume that new constants introduced in the restriction and scope of the quantifier are invariably passed along to the output coordinate C. All of this information can be expressed grammatically by suitably annotating the syntactic category associated with the determiner, quantifier, or connective in question and using this information to constrain the composition of context-dependent interpretations.

Although the empirical properties of natural language quantificational phenomena are complex and often vexed, I have tried to show here how the basic framework advanced here allows the insights of research on dynamic semantics to be integrated into a system of labeled deduction.

3 Taking stock

In the sections above, I have tried to show how theories of dynamic interpretation can be integrated with systems of labeled deduction in a uniform way and that the picture of binding, quantification, and anaphora that results can be described in part by finite-state methods. This picture is not yet complete. Nevertheless, since the general framework allows the simulation of a broad range of binding theories, it offers a means of comparing alternative accounts. At the same time, it makes it possible to try to extract the finite-state aspects of binding phenomena from other properties.

There are many other issues in this domain which deserve to be explored. Among the most obvious are:

- possible connections with Van Benthem's work on semantic automata [1];
- the study of natural language predicates and regular relations, especially with regard to plurality [13].

I hope to be able to pursue these connections in future work.

REFERENCES

[1] J. van Benthem. 1986. *Essays in Logical Semantics.* D. Reidel. Dordrecht.

[2] M. Brame. 1977. Alternatives to the Tensed-S and Specified Subject Conditions. *Linguistics and Philosophy* **1**.381-411.

[3] N. Chomsky. 1982. *Some concepts and consequences of the theory of Government and Binding.* Linguistic Inquiry Monograph Six. MIT Press.

[4] D. Gabbay and Ruy de Queiroz. 1992. Extending the Curry-Howard Interpretation. *Journal of Symbolic Logic.*

[5] J.-Y. Girard, Y. Lafont, and P. Taylor. 1989. *Proofs and Types.* Cambridge University Press. Cambridge.

[6] J. Groenendijk & M. Stokhof. 1991. Dynamic predicate logic. *Linguistics & Philosophy* **14**.39-100.

[7] J. Groenendijk & M. Stokhof. 1991. Dynamic Montague Grammar. J. Groenendijk, M. Stokhof, and D. Beaver, eds., *Quantification and anaphora I.* DYANA report R2.2.A, Edinburgh. Centre for Cognitive Science, University of Edinburgh.

[8] H. Kamp & U. Reyle. 1993. *From Discourse to Logic.* Kluwer. Dordrecht.

[9] R. Kempson. 1995. Crossover: a dynamic perspective. In S. Jensen, ed., *SOAS Working Papers in Linguistics and Phonetics 6.* SOAS. London.

[10] M. Moortgat. 1995. In situ binding: a modal analysis. *Proceedings of the 10th Amsterdam Colloquium.* ILLC. Universiteit van Amsterdam. Amsterdam.

[11] R. Oehrle. 1995. Some 3-dimensional systems of labeled deduction. *Bulletin of the Interest Group in Pure and Applied Logic.* **3**.2-3.

[12] R. Oehrle. 1995. Labeled deduction, Binding, Quantification. Symposium on Proof Theory and Natural Language, Barcelona. 25 page manuscript.

[13] R. Oehrle. 1996. Austinian Pluralities. In J. Seligman and D. Westerståhl, eds., *Logic, Language, and Computation.* CSLI Lecture Notes. Cambridge University Press. Cambridge.

[14] C. Pollard & I. Sag. 1992. Anaphors in English and the scope of binding theory. *Linguistic Inquiry* **23**.261-303.

[15] R. Thomason. 1974. *Formal Philosophy: Selected Papers of Richard Montague*, Yale University Press, New Haven.

[5] Kempson [9] provides a much more thoroughgoing analysis of crossover phenomena from a point of view similar to that advocated here.