Combining Montague Semantics and Discourse Representation

I. Introduction

Kamp's Discourse Representation Theory (DRT, see Kamp [1981], Heim [1982, 1983], Kamp & Reyle [1993]) offers an attractive means of modeling the semantics of discourse. The theory works very well for cross-sentential pronominal anaphora, for temporal anaphora and, on the sentence level, for the medieval 'donkey' sentences that Geach has reminded us about so forcefully. However, other classes of data, such as quantification and coordination phenomena, are handled in less elegant ways, and since such data were rather successfully accounted for within Montague's earlier paradigm, some logicians have sought to combine the advantages of both approaches and have come up with various systems that combine the dynamics of DRT with the compositionality of Montague Semantics in one way or another.

There are two lines of approach here. Some authors start with a Montague-like set-up and add extra machinery to it in order to deal with anaphora; others take DRT as their point of departure and add lambdas to the system in order to obtain compositionality. An example of the first line of thought is Rooth [1987], who extends Montague Grammar with the 'parameterised sets' of Barwise [1987]. Other examples are Groenendijk & Stokhof [1990], who obtain a dynamic effect with the help of Janssen's [1986] 'state switchers'; Chierchia [1992], who gives a simplified variant of Groenendijk & Stokhof's 'Dynamic Montague Grammar' (DMG); and Dekker [1993], who elaborates upon DMG and extends it with many useful features. The second line of approach is exemplified in the λ-DRT framework of Pinkal and his co-workers (Latecki & Pinkal [1990], Bos et al. [1994]) and in the 'bottom-up' DRT of Asher [1993].

However, while all this work has certainly clarified the issues which are at stake when an integration of the two semantic frameworks is called for, and while some of this work has led to the development of formalisms in which very interesting linguistic theories have been expressed, we are still not in the possession of a combined framework which is easy to use and mathematically clean at the same time. The approaches which add lambdas to DRT are easy to use, while the systems which take Montague Semantics as their point of departure are difficult to work with. All systems base themselves on *ad hoc* special purpose logics and have a more or less baroque underlying mathematics which is not very well understood.

---

* From: *Linguistics and Philosophy* **19**: 143-186, 1996. I would like to thank my audience at the Ninth Amsterdam Colloquium and the anonymous referees for interesting feedback. The present title was kindly suggested by an anonymous referee.

It is the purpose of this paper to combine Montague Semantics and Discourse Representation into a formalism that is not only notationally adequate, in the sense that the working linguist need remember only a few rules and notations, but is also mathematically rigourous and based on ordinary type logic. The mathematics of lambdas is well-known and has received a classical formulation in the elegant system of Church [1940] and for the present purposes I hardly see any reason to tamper with the logic. Later extensions of Church's logic, such as Montague's IL, but also Janssen's [1986] Dynamic Intensional Logic, upon which DMG is based, add all kinds of embellishments, but these make the logic lose its simplicity and its fundamental mathematical properties.[1] Usually, the embellishments are superfluous, as Church's logic is already very expressive so that what can be said with embellishments can also be said without them.[2] It is for this reason that we shall stick to (a many-sorted variant of) the classical system. Our goal is to design a formalism for linguistic description which is easy to use in practice and has a simple underlying mathematics.

How can we combine the DRT logic with classical type theory? It is evident that the expressivity of the core part of DRT does not extend that of first-order logic (Kamp & Reyle [1993] in fact give a translation of DRT into elementary logic). The question therefore arises whether it is not possible to *reduce* the language of DRT to type theory. I shall show that this question has an affirmative answer and that DRT's Discourse Representation Structures (DRSs or boxes henceforth) are already present in classical type logic in the sense that they can simply be viewed as abbreviations of certain first-order terms, provided that some first-order axioms are adopted. This means that we can have boxes and lambdas in one logic, and the combination of these two (plus the sequencing operator of Dynamic Logic, which is definable in type logic as well) will allow us to assign boxes to English discourses in accordance with Frege's Principle: the meaning of a complex expression is a function of the meanings of its parts.

The method of showing that the DRS language is really a part of type logic (and even of predicate logic) is straightforward. Boxes are connected to first-order models via 'verifying embeddings' in Kamp's definition and in fact the meaning of a box can be viewed as a binary relation between such embeddings. Groenendijk & Stokhof [1991] give an ordinary Tarski Definition for the box language on the basis of this idea. The reduction to classical logic can be carried out by simply observing that the right-hand, *definiens,* parts of the clauses in this inductive definition are expressible within the logic and that the constructs in the box language that are defined by them

---

[1] Such as the Church-Rosser property, which says that the order in which lambda reductions are performed is immaterial. See Friedman & Warren [1980] for a counterexample to the Church-Rosser property in IL.

[2] For example, Gallin [1975] already shows that what can be said within IL can be said within the Church system TY_2.

may therefore do duty as abbreviations of the logical terms which express these clauses.

The presence of boxes in type logic permits us to fuse DRT and Montague Grammar in a rather evenhanded way: both theories will be recognisable in the result. In Muskens [1991, 1995a] I have given versions of Montague Grammar that were based on some technical insights that are also present in this paper, but while these articles succeed in importing the dynamics of DRT into Montague Grammar, they do not offer a real synthesis of the two theories in this sense. The present formalism, which I call Compositional DRT (CDRT), may well be described as a version of DRT in which the construction algorithm that sends (parsed) sentences to boxes consists of a Montague style compositional translation of trees into the lambda-box language.

With this unification of the theories standard techniques (such as type-shifting) that are used in Montague Grammar become available in DRT. The fused theory gives us a means to compare specific semantic analyses that are made in the two different frameworks and semantic theories that can be expressed within the older paradigm can in principle be transposed to our version of the newer one. Although descriptive linguistic work is not the principal aim of this paper, we provide an illustration here and it will be shown how the Boolean theory of generalised coordination can be adapted to Compositional DRT. Various authors (e.g. Von Stechow [1974], Keenan & Faltz [1978], Gazdar [1980], Partee & Rooth [1983]) have suggested that, barring some exceptions, the words `and` and `or` and their counterparts in other languages act as Boolean operators, no matter what types of expressions they connect. This, I think, is a very elegant and empirically significant claim about natural languages. Unfortunately, it does not seem to match completely with the data, as it does not seem possible to treat expressions with anaphoric links across coordinated elements (such as in *A cat catches a fish and eats it* vs. **A cat catches no fish and eats it*) in a Boolean way. On the other hand it must be admitted that the treatment in Kamp & Reyle [1993] of such expressions, while satisfactory in the sense that it accounts for the anaphoric linkings, is unsatisfactory in the sense that it complicates the whole set-up of DRT. Discourse Representations are no longer boxes, but complicated constructs out of boxes and natural numbers in this analysis; even the treatment of expressions that do not contain a conjunction cannot escape from the complications. It would be much nicer if we could connect a single algebraic operation to each of the two words `and` and `or`. In section IV we shall provide such operations.

The paper is set up as follows. In the next section standard DRT is discussed and provided with some extensions, such as the sequencing operator ';', which is familiar from programming languages, and a compositional treatment of proper names as constants. In section III, the main part of the paper, we define the basic formalism and show how it can be used to interpret a fragment of English that is generated with the help of a reduced transformational grammar. Section IV extends this fragment with the promised treatment of coordinations and section V gives conclusions and further prospects. In a short appendix the reader finds two proofs of propositions which were mentioned in the main text.

II. STANDARD DRT AND SOME EXTENSIONS

II.1. The core fragment: syntax and semantics

The standard way of writing down discourse representations makes for very easy reading but is also rather space consuming and for this reason I shall linearize DRS boxes in this paper. So, for example, (2), a representation of the little text in (1), will be written more concisely as (3) below, and I shall save paper by writing (6) instead of the more wasteful (5), which is a representation of (4).

(1) A man adores a woman. She abhors him.

(2)
$$\boxed{\begin{array}{c} x_1 \quad x_2 \\ \\ \mathit{man}\ x_1 \\ \mathit{woman}\ x_2 \\ x_1\ \mathit{adores}\ x_2 \\ x_2\ \mathit{abhors}\ x_1 \end{array}}$$

(3) $[x_1\ x_2 \mid \mathit{man}\ x_1,\ \mathit{woman}\ x_2,\ \ x_1\ \mathit{adores}\ x_2,\ x_2\ \mathit{abhors}\ x_1]$

(4) If a man bores a woman she ignores him.

(5)
$$\boxed{\begin{array}{c} \\ \boxed{\begin{array}{c} x_1 \quad x_2 \\ \\ \mathit{man}\ x_1 \\ \mathit{woman}\ x_2 \\ x_1\ \mathit{bores}\ x_2 \end{array}} \Rightarrow \boxed{x_2\ \mathit{ignores}\ x_1} \\ \\ \end{array}}$$

(6) $[\ \mid [x_1\ x_2 \mid \mathit{man}\ x_1,\ \mathit{woman}\ x_2,\ x_1\ \mathit{bores}\ x_2] \Rightarrow [\ \mid x_2\ \mathit{ignores}\ x_1]]$

Formally, the language which comprises these linearized forms is obtained by starting with sets of relation constants and individual constants and an infinite set of individual variables (individual constants and variables will also be called *discourse referents*). Conditions and boxes are constructed from these by the following clauses.

SYN1 If R is an n-ary relation constant and $\delta_1,\ldots,\delta_n$ are discourse referents, then $R(\delta_1,\ldots,\delta_n)$ is a condition;
If δ_1 and δ_2 are discourse referents then δ_1 **is** δ_2 is a condition;
SYN2 If K_1 and K_2 are boxes, **not** K_1, K_1 **or** K_2 and $K_1 \Rightarrow K_2$ are conditions;
SYN3 If $\gamma_1,\ldots,\gamma_m$ are conditions ($m \geq 0$) and $x_1,\ldots,x_n$ are variables ($n \geq 0$), then $[x_1\ldots x_n \mid \gamma_1,\ldots,\gamma_m]$ is a box.

That the language given here contains *constant* discourse referents in addition to the usual variable ones is a novelty which has to do with the interpretation of proper names. This will be discussed below. In (3) and (6) we have systematically abbreviated conditions of the form $R(\delta_1,\delta_2)$ as $\delta_1 R \delta_2$ in case R was a two-place relation symbol and we have written $R(\delta)$ as $R\delta$ whenever R was one-place. We shall continue to follow this convention.

The constructs generated by SYN1-SYN3 are interpreted on ordinary first-order models. As usual, these are defined as pairs $\langle D, I\rangle$, where D is an arbitrary non-empty set and I is a function with the set of constants as its domain such that $I(c) \in D$ for each individual constant c and $I(R) \subseteq D^n$ for each n-ary relation constant R. An *assignment,* or, in DRT terminology, an *embedding,* for such a first-order model $M = \langle D, I\rangle$ is a function from the set of variable discourse referents to the domain D. We write $a[x_1\ldots x_n]a'$ as an abbreviation for 'assignments a and a' differ at most in their values for $x_1,\ldots,x_n$'. As usual, we let $\|\delta\|^{M,a}$ be equal to $a(\delta)$ if δ is a variable and to $I(\delta)$ if δ is a constant. Clauses SEM1–SEM3 below define the semantic value $\|\gamma\|^M$ of a condition γ in a model M to be a set of embeddings, while the value $\|K\|^M$ of a box K in M is defined to be a binary relation between embeddings. (The superscript M is suppressed.)

SEM1	$\|R(\delta_1,\ldots,\delta_n)\|$	=	$\{a \mid \langle\|\delta_1\|^a,\ldots,\|\delta_n\|^a\rangle \in I(R)\}$
	$\|\delta_1$ **is** $\delta_2\|$	=	$\{a \mid \|\delta_1\|^a = \|\delta_2\|^a\}$
SEM2	$\|$**not** $K\|$	=	$\{a \mid \neg\exists a' \langle a, a'\rangle \in \|K\|\}$
	$\|K_1$ **or** $K_2\|$	=	$\{a \mid \exists a' (\langle a, a'\rangle \in \|K_1\| \vee \langle a, a'\rangle \in \|K_2\|)\}$
	$\|K_1 \Rightarrow K_2\|$	=	$\{a \mid \forall a'(\langle a, a'\rangle \in \|K_1\| \rightarrow \exists a''\langle a', a''\rangle \in \|K_2\|)\}$
SEM3	$\|[x_1\ldots x_n \mid \gamma_1,\ldots,\gamma_m]\|$	=	$\{\langle a, a'\rangle \mid a[x_1\ldots x_n]a' \;\&\; a' \in \|\gamma_1\| \cap\ldots\cap \|\gamma_m\|\}$

The interpretation here may look somewhat different from the standard interpretation of DRT (as, for example it is given in Kamp & Reyle [1993]), but is in fact equivalent.[3] For a discussion of this slightly different format as compared to the standard one I refer to Groenendijk & Stokhof [1991].[4]

A box K is defined to be *true* in a model M under an embedding a if and only if there is some embedding a' such that $\langle a, a' \rangle \in \|K\|^M$; a condition γ is *true* in M under a iff $a \in \|\gamma\|^M$. The reader is invited to verify that the truth-conditions of (3) above correspond to the truth-conditions of the more conventional (7), and that the truth-conditions of (6) are those of (8).

(7) $\exists x_1 x_2(man\ x_1 \wedge woman\ x_2 \wedge x_1\ adores\ x_2 \wedge x_2\ abhors\ x_1)$

(8) $\forall x_1 x_2((man\ x_1 \wedge woman\ x_2 \wedge x_1\ bores\ x_2) \rightarrow x_2\ ignores\ x_1)$

Boxes can be interpreted as instructions to *change* the current embedding in a non-deterministic way. For example, given some input embedding a, box (3) can be read as an instruction to output an arbitrary embedding a' such that $a[x_1,x_2]a'$, while $a'(x_1)$ is a man, $a'(x_2)$ is a woman, $a'(x_1)$ adores $a'(x_2)$ and $a'(x_2)$ abhors $a'(x_1)$. If it is possible to find such an embedding a' then (3) is true.

As many authors (e.g. Van Benthem [1989, 1991], Groenendijk & Stokhof [1991]) have emphasized, there is a strong connection with Quantificational Dynamic Logic (Pratt [1976]), the logic of computer programs, here. In fact it can be shown that DRT is a fragment of this logic. For a discussion of this last point see Muskens [1995a] or Muskens, Van Benthem & Visser [forthcoming].

II.2. Merging boxes

In standard Discourse Representation Theory sentences in discourse do not in general get a separate interpretation each. Let us reconsider text (1), represented here as (9), with its indefinite determiners and pronouns now indexed in order to show the intended anaphoric relationships.

(9) A^1 man adores a^2 woman. She$_2$ abhors him$_1$.

---

[3] Since we have enlarged the language with constant discourse referents the equivalence is to be restricted to constructs that do not contain these.

[4] The main difference with the standard DRT set-up is that we use total assignments, while standard DRT uses partial ones. Although it is immaterial from a formal point of view which set-up is chosen as long as our purpose is limited to giving a truth definition for DRT, it may very well be that the partial approach is better inasfar as it is more realistic and therefore more liable to suggest useful generalisations of the theory. I leave a partialisation of the theory presented in this paper for future research. For a partial theory of types on which such a partialisation could be based see Muskens [1989, 1995b]. A second difference with standard DRT lies in the treatment of cases such as $[\ |\ [x \mid man\ x] \Rightarrow [x \mid mortal\ x]]$, where x is declared twice. In standard DRT the second declaration of x is ignored and the box would express that every man is mortal. In the present set-up the second x assumes a new value and the box says that there is a mortal if there is a man. The construction algorithm for standard DRT never generates such cases of 'reassignment', but we shall see an application in section IV below.

While the first sentence in this little text can indeed be said to have its own representation, namely (10), the second is not associated with a box itself in the standard approach. Its contribution to the Discourse Representation that will finally be built lies in the transition from (10) to (3). But if, as is our purpose in this paper, we want the meaning of a complex text to be a function of the meanings of its constituing sentences, we must indeed assign separate meanings to the latter.

The only reasonable box that we can associate with the open sentence *She*$_2$ *abhors him*$_1$ is the open box (11) below which is true under an embedding a if and only if the condition x_2 *abhors* x_1 is true under a. The anaphoric pronouns *she*$_2$ and *him*$_1$ get whatever value the input embedding associates with the discourse referents x_2 and x_1. Box (11) can be interpreted as a *test*: given any input embedding a, it tests whether $a(x_2)$ abhors $a(x_1)$, if so, it returns a as an output, if not, the test fails and no output is returned.

(10) $[x_1\ x_2 \mid man\ x_1,\ woman\ x_2,\ x_1\ adores\ x_2]$
(11) $[\ \mid x_2\ abhors\ x_1]$

How can we combine boxes (10) and (11)? The metaphor that invites us to view boxes as programs suggests an answer: first carry out the instruction (10) (non-deterministically) and then carry out (11). In other words, the metaphor suggests that the concatenation of sentences in simple narrative discourse is nothing but the *sequencing* of instructions that we find in imperative programming languages, usually written as ';'. This would mean that (12) would be the correct translation of text (9).

(12) $[x_1\ x_2 \mid man\ x_1,\ woman\ x_2,\ x_1\ adores\ x_2]\ ;\ [\ \mid x_2\ abhors\ x_1]$

In order to make this a legal expression of our language let us extend the DRT syntax with the sequencing operator.

SYN4 If K_1 and K_2 are boxes then $K_1 ; K_2$ is a box

In contexts where programs are modeled as relations between states, sequencing is usually associated with the operation of relational composition. A program $K_1 ; K_2$ can bring the machine from an input state a to an output state a' if and only if K_1 can bring it from a to some intermediary state a'' and K_2 can take it from a'' to a'. In this paper we shall also associate relational composition with the sequencing of boxes as clause SEM4 makes clear.

SEM4 $\|K_1 ; K_2\| \quad = \quad \{\langle a, a'\rangle \mid \exists a'' (\langle a, a''\rangle \in \|K_1\| \ \&\ \langle a'', a'\rangle \in \|K_2\|)\}$

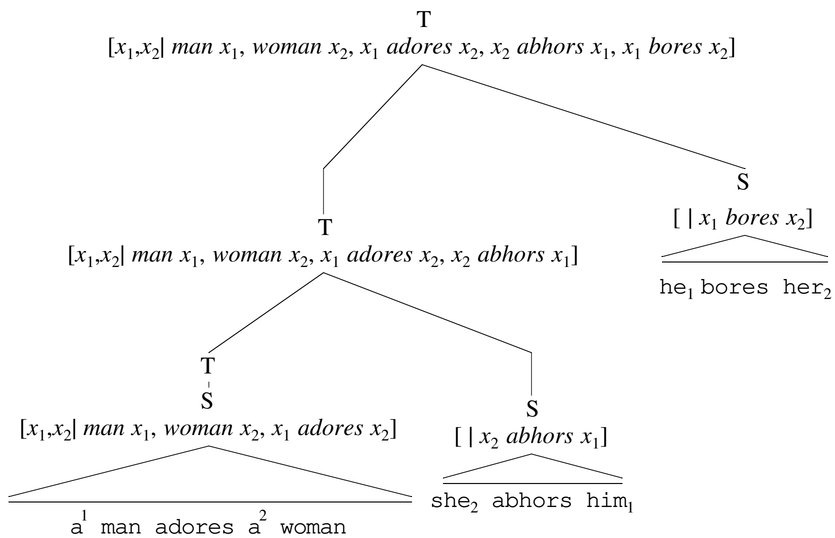
It is easy to verify that (12) and (3), our new translation of the text in (9) and the old one, are equivalent. More in general, we have the following lemma, which has a simple proof.

MERGING LEMMA. If $x'_1,\ldots,x'_k$ do not occur in any of $\gamma_1,\ldots,\gamma_m$ then
$\|[x_1\ldots x_n \mid \gamma_1,\ldots,\gamma_m] \,;\, [x'_1\ldots x'_k \mid \delta_1,\ldots,\delta_q]\| = \|[x_1\ldots x_n x'_1\ldots x'_k \mid \gamma_1,\ldots,\gamma_m, \delta_1,\ldots,\delta_q]\|$

For obvious reasons $[x_1\ldots x_n x'_1\ldots x'_k \mid \gamma_1,\ldots,\gamma_m, \delta_1,\ldots,\delta_q]$ is usually called the *merge* of boxes $[x_1\ldots x_n \mid \gamma_1,\ldots,\gamma_m]$ and $[x'_1\ldots x'_k \mid \delta_1,\ldots,\delta_q]$.

In the presence of the sequencing operator we can provide texts with compositional translations, as (13) illustrates.

(13)



But clearly, given the tools we have developed thus far, this can only work on a suprasentential level. For compositionality on the subsentential level we need an additional strategy, which will be developed in section III.

II.3. Proper Names

The treatment of proper names in DRT seems to be inherently non-local inasfar as the proper name rule in the DRS construction algorithm places material in the topmost box of the DRS under construction. In the earliest version of the theory (Kamp [1981]), for example, sentence (14) would be translated as (15). Even though the names ‘Sue’ and ‘Tom’ are encountered in the antecedent of the conditional, the discourse referents x_1 and x_2 they give rise to are placed at the top level of the discourse representation structure. Conditions x_1 **is** *sue* and x_2 **is** *tom*, where *sue* and *tom* are individual constants, constrain the interpretation of these discourse referents and are also placed at top level. In more recent versions of the theory (such as Kamp & Reyle [1993]) we find a slightly different form for these constraining conditions:

(16) is the DRS that is connected with (14) now. The expressions *Sue* and *Tom* are one-place predicate symbols here, denoting the predicates 'being a Sue' and 'being a Tom', or perhaps 'being named Sue' and 'being named Tom'. This means that situations in which more than one person carries a particular name are allowed for now. But in this approach too, we find that discourse referents connected with names and the conditions constraining them are put at the top of the box, not in the antecedent of the conditional.

(14) If Sue ignores Tom he is miserable. He adores her.

(15) $[x_1\ x_2 \mid x_1\ \mathbf{is}\ sue,\ x_2\ \mathbf{is}\ tom,\ [\ \mid x_1\ ignores\ x_2] \Rightarrow [\ \mid miserable\ x_2],$
$x_2\ adores\ x_1]$

(16) $[x_1\ x_2 \mid Sue\ x_1,\ Tom\ x_2,\ [\ \mid x_1\ ignores\ x_2] \Rightarrow [\ \mid miserable\ x_2],$
$x_2\ adores\ x_1]$

However, as Kamp & Reyle (pp. 246-248) note, the interpretation strategy which is illustrated by (16) is wrong. In a situation where two girls a and b are both called 'Zebedea' we cannot use 'Zebedea loves a stockbroker' to express that either a or b loves a stockbroker; the sentence can only be used with unique reference to some Zebedea. For this reason Kamp & Reyle propose to adopt the device of *external anchoring*. An external anchor is a finite function from discourse referents to the objects they are meant to denote and in Kamp & Reyle's proposal these anchors may appear in the discourse representation. For example, if d_1 is Sue and d_2 is Tom, then (17) is now a DRS for (14).

(17) $[x_1\ x_2 \mid \{\langle x_1, d_1\rangle, \langle x_2, d_2\rangle\},\ [\ \mid x_1\ ignores\ x_2] \Rightarrow [\ \mid miserable\ x_2],$
$x_2\ adores\ x_1]$

The idea is that x_1 and x_2 are constrained to be Sue and Tom respectively. Formally, we can let an anchor f be a condition with interpretation $\{a \mid f \subseteq a\}$, so that the interpretation of (17) becomes the set of all pairs $\langle a, a'\rangle$ such that a and a' differ at most in x_1 and x_2, $a'(x_1) = d_1$ (= Sue), $a'(x_2) = d_2$ (= Tom) and such that a' also satisfies the last two conditions of (17).

Two remarks can be made. The first is that we seem to have an illustration of the non-transitivity of improvement here: although the interpretation strategy that gave rise to (17) was presented as an improvement over the strategy that lead to (16), and (16) as an improvement over (15), we cannot conclude that (17) is an improvement over (15) because, in the intended models where $I(sue) = d_1$ and $I(tom) = d_2$, these boxes have exactly the same denotations, and in fact it can be said that the external anchor $\{\langle x_1, d_1\rangle, \langle x_2, d_2\rangle\}$ is just a way to express what would otherwise be expressed as x_1 **is** *sue*, x_2 **is** *tom*. A second remark is that discourse markers which are anchored to specific objects in the model in fact are no variables but are *constants*. If x_1 and x_2 are anchored to Sue and Tom from the outset, these variables are not allowed to vary and in fact function as constants; but then why not straightforwardly use constants *sue* and *tom* instead, which are already appropriately anchored by the interpretation function? The intended translation of (14) then becomes (18).

(18) [| [| *sue ignores tom*] ⇒ [| *miserable tom*], *tom adores sue*]

In order for this to work we must allow constants to be discourse referents, as we have done in II.1. A name will simply be translated as a constant discourse referent and pronouns can pick up constant discourse referents from any position, provided that these were introduced somewhere in the text.

An important advantage of this strategy is that it is no longer necessary to have a non-local rule for the interpretation of names: since constants are scopeless we can interpret them *in situ*. This solves a difficulty that compositional treatments of DRT, such as Groenendijk & Stokhof's DPL and DMG systems and the systems in Muskens [1991, 1995a], usually have with proper names. The special form of the DRT rule for processing names is an artefact of the standard convention that only variables can function as discourse referents. As soon as we allow constants to be discourse referents as well, we can process names locally. Although this procedure is optically different from the approach where discourse markers for names are externally anchored, there seems to be no real difference, as externally anchored referents in fact *are* constants.

III. COMPOSITIONAL DRT

III.1. The Logic of Change

The sequencing operator introduced in II.2 allows us to compute the meaning of certain texts from the meanings of the sentences they are built from, but it does not allow us to compute the meanings of these sentences from those of smaller building blocks of language. The theory simply does not assign meanings to such constituents. If we want to build up meanings compositionally from the lexical level to the level of texts we must provide for representations of such meanings and we shall therefore adopt Montague's strategy to introduce lambda-abstraction and application into the logical language. In this section and the next ones we shall see how these two logical notions can be combined with the DRT syntax that was defined above. Fortunately there is no need to do anything very special on the logical side, for it turns out that ordinary many-sorted type logic satisfies our needs. There is a natural way to emulate the DRT language in type logic, provided that we adopt a few axioms. As we have seen, the DRT language talks about embeddings and our set-up will enable us to do likewise.

We shall have at least four types of primitive objects in our logic:[5] apart from the ordinary cabbages and kings sort of entities (type *e*) and the two truth values (type *t*) we shall also allow for what I would like to call *pigeon-holes* or *storages* or *registers* (type π) and for *states* (type *s*). Registers, which are the things that are denoted by discourse referents, may be thought of as small chunks of space that can contain exactly one object. The intuitive idea (which should not be taken too seriously, however, since we are talking about non-deterministic processes) is that whenever in a text we encounter an indefinite noun phrase like `a pigeon`, some pigeon is stored in a register connected to the indefinite; whenever we encounter the pronoun `it`, anaphorically related to the indefinite, we interpret it as referring to the contents of that register. A proper name like `Sue` will also be connected with a register and a pronoun `she` anaphorically related to `Sue` will be interpreted as the contents of that storage. This explains how in a little text such as (19) the anaphoric relationships are established.

(19) $\texttt{Sue}^{3}$ has $\texttt{a}^{8}$ pigeon. $\texttt{She}_{3}$ feeds $\texttt{it}_{8}$.

There is a difference in kind between registers connected to indefinites and registers connected to names. The contents of the first can always be changed and will accordingly be called *variable registers*. The second kind of registers always have a fixed inhabitant and are called *constant registers.* Variable registers and constant registers will very much play the role that individual variables and constants played in section II, but unlike these they are model-theoretic objects. Discourse referents will simply be *names* of registers, i.e. constants of type π. Those naming variable registers will be called *unspecific discourse referents*; those denoting constant registers are called *specific discourse referents.* Note that an unspecific discourse referent is a constant of the language itself, but denotes an object which may be thought of as a variable; a specific discourse referent is a constant which denotes something which always has the same value.

A *state* may be thought of as a list of the current inhabitants of all registers. States are very much like the program states that theoretical computer scientists talk about, which are lists of the current values of all variables in a given program at some stage of its execution.

We will typically use *u* to range over unspecific referents, but *Tim, Tom, Mary* (written with an initial capital letter) and the like to range over specific ones. Variables over registers are possible as well and we shall typically use *v* for these. Variables of type *e* are typically denoted with *x* and we shall also retain the type *e* constants *tim,*

---

[5] For each *n* let us define TY_n just as the logic TY_2 in Gallin [1975], but with *n* basic types other than *t* instead of Gallin's two types *e* and *s.* TY_0 is the logic described in Henkin [1963], TY_1 is Church's [1940] original theory of types, TY_2 is Gallin's logic, TY_3 is the logic that we are working in presently. Of course, all systems in this hierarchy, with the exception of TY_0 (in which all Henkin models are standard models and vice versa), have virtually the same metatheory.

tom, mary (no capital) from section II. Table 1 sums up the various conventions that will be used.

Type	**Name of objects**	**Variables**	**Constants**
s	states	i, j, k, h	
e	entities	$x_1, x_2, \ldots$	*mary*, …
π	registers	v	$u_1, u_2, \ldots$ (unspecific discourse referents) *Mary*, … (specific discourse referents)

Table 1.

Pictorially the relation between states, entities and registers is as in Figure 1. We have an infinity of unspecific referents $u_1,\ldots,u_n,\ldots$, a set of specific referents *Tim, Tom, Mary,…* and an infinity of states $i_1,\ldots,i_n,\ldots$. Each discourse referent corresponds to a register which in each state has some occupant (an entity). We shall let V be a fixed non-logical constant of type $\pi(se)$ and denote the inhabitant of register δ in state i with the type e term $\mathsf{V}(\delta)(i)$. In the given figure $\mathsf{V}(u_3)(i_4)$ = Lee for example. Note that each state i corresponds to a function $\lambda v(\mathsf{V}(v)(i))$, which is as close to an embedding as one can get, since it assigns an entity to each variable register.

In order to impose the necessary structure on our models we must adopt some definitions and axioms. First a definition: let $i[\delta_1\ldots\delta_n]j$ be short for

$$\forall v((\delta_1 \neq v \wedge \ldots \wedge \delta_n \neq v) \rightarrow \mathsf{V}(v)(i) = \mathsf{V}(v)(j)),$$

	i_1	i_2	i_3	i_4	i_5	$\cdots$
u_1:	Bob	Joe	Joe	Tim	Tom	$\cdots$
u_2:	Tim	Jim	Ann	Sue	Rob	$\cdots$
u_3:	Lee	Bob	Lee	Lee	Jan	$\cdots$
u_4:	Sue	Pat	Sue	Sue	Jan	$\cdots$
u_5:	Ann	Ann	Ann	Ann	Sue	$\cdots$
u_6:	Tom	Ann	Tom	Tom	Jim	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
Tim:	Tim	Tim	Tim	Tim	Tim	$\cdots$
Joe:	Joe	Joe	Joe	Joe	Joe	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Figure 1

for all type s terms i and j and all $\delta_1,\ldots,\delta_n$ of type π; $i[]j$ will stand for the formula $\forall v(\mathsf{V}(v)(i) = \mathsf{V}(v)(j))$. The formula $i[\delta_1\ldots\delta_n]j$ now expresses that i and j differ at most in $\delta_1,\ldots,\delta_n$. Next the axioms. Letting VAR be a predicate of type πt (singling out the variable registers), we require the following:

AX1 $\forall i\forall v\forall x(\mathsf{VAR}(v) \rightarrow \exists j(i[v]j \wedge \mathsf{V}(v)(j) = x))$
AX2 $\mathsf{VAR}(u)$, if u is an unspecific referent
AX3 $u_n \neq u_m$, for each two different unspecific referents u_n and u_m
AX4 $\forall i(\mathsf{V}(Tom)(i) = tom)$,
$\forall i(\mathsf{V}(Mary)(i) = mary)$,
$\forall i(\mathsf{V}(Tim)(i) = tim)$, etc., for all names in the fragment.

AX1 demands that for each state, each variable register and each individual, there must be a second state that is just like the first one, except that the given individual is an occupant of the given register. The axiom is connected to Janssen's [1986] 'Update Postulate' and to 'Having Enough States' in Dynamic Logic (see e.g. Goldblatt [1987]). The second axiom says that unspecific referents refer to variable registers and the third axiom demands that different unspecific discourse referents denote different registers. This is necessary, as an update on one discourse referent should not result in a change in some other discourse referent's value. The fourth axiom scheme ensures that constant registers always have the same inhabitant and establishes the obvious connection between constant referents and the corresponding constants of type e.

Type logic enriched with these first-order non-logical axioms (axiom schemes) will be our 'Logic of Change'. The logic has the very useful property that it allows us to have a form of the 'unselective binding' that seems to be omnipresent in natural language (see Lewis [1975]). Since states (Lewis's 'cases'[6]) correspond to lists of items, a single quantification over states may correspond to multiple quantifications over the items in such a list. The following lemma gives a precise formulation of this phenomenon; it has an elementary proof based on AX1 and AX2.

UNSELECTIVE BINDING LEMMA. Let $u_1,\ldots,u_n$ be unspecific referents of type π, let $x_1,\ldots,x_n$ be distinct variables of type e, let φ be a formula that does not contain j and write $[\mathsf{V}(u_1)(j) / x_1,\ldots,\mathsf{V}(u_n)(j) / x_n]\varphi$ for the simultaneous substitution of $\mathsf{V}(u_1)(j)$ for x_1 and … and $\mathsf{V}(u_n)(j)$ for x_n in φ, then:

$$\models_{\mathrm{AX}} \forall i(\exists j(i[u_1,\ldots,u_n]j \wedge [\mathsf{V}(u_1)(j) / x_1,\ldots,\mathsf{V}(u_n)(j) / x_n]\varphi) \leftrightarrow \exists x_1\ldots\exists x_n\varphi)$$

6 The items in our lists are all of individual type, whereas Lewis's cases are lists of objects of various types. See Muskens [1995a] for a generalisation that will allow objects of many types to be the inhabitants of registers.

III.2. Boxes in type logic

The reader may note that all concepts expressed in the language of set theory that were needed to give a semantics for DRT in clauses SEM1–SEM4 are now available in our type logic. This means that it is possible now to have analogues of these clauses as abbreviations in our logic. In particular (letting δ typically range over terms of type π, specific as well as unspecific discourse referents) we may agree to write

ABB1	$R\{\delta_1,\ldots,\delta_n\}$	for	$\lambda i.R(\mathsf{V}(\delta_1)(i))\ldots(\mathsf{V}(\delta_n)(i))$,
	δ_1 **is** δ_2	for	$\lambda i.\mathsf{V}(\delta_1)(i) = \mathsf{V}(\delta_2)(i)$,
ABB2	**not** K	for	$\lambda i\neg\exists jK(i)(j)$,
	K **or** K'	for	$\lambda i\exists j(K(i)(j) \vee K'(i)(j))$,
	$K \Rightarrow K'$	for	$\lambda i\forall j(K(i)(j) \rightarrow \exists kK'(j)(k))$,
ABB3	$[u_1\ldots u_n \mid \gamma_1,\ldots,\gamma_m]$	for	$\lambda i\lambda j(i[u_1,\ldots,u_n]j \wedge \gamma_1(j) \wedge\ldots\wedge \gamma_m(j))$,
ABB4	$K\,;K'$	for	$\lambda i\lambda j\exists k(K(i)(k) \wedge K'(k)(j))$.

In ABB1 R must be a constant of type e^nt, where e^nt is defined by letting e^0t be t and $e^{k+1}t$ be $e(e^kt)$. We shall write $R\{\delta_1,\delta_2\}$ as $\delta_1 R\delta_2$ and $R\{\delta_1\}$ as $R\delta_1$.

The principal difference between the clauses given here and SEM1–SEM4 is that we no longer *interpret* the boxes and conditions of DRT in a metalanguage, but that we consider them to be *abbreviations* of expressions in our type logical object language. Boxes are abbreviations of certain $s(st)$ terms, conditions are shorthand for terms of type st. Otherwise SEM1–SEM4 and ABB1–ABB4 are much the same.

Let us see how these abbreviations work. First note that e.g. u_2 *abhors John* turns out to be short for $\lambda i.abhors(\mathsf{V}(u_2)(i))(\mathsf{V}(John)(i))$, which, given AX4, is equivalent to $\lambda i.abhors(\mathsf{V}(u_2)(i))(john)$. The discourse referents u_2 and *John* have an indirect reference here: they act as addresses of the memory locations where, in state i, the objects $\mathsf{V}(u_2)(i)$ and $\mathsf{V}(John)(i)$ are to be found. Next, consider (20); ABB3 tells us that this term is really short for (21) and with ABB1 plus λ-conversion we find that the latter is equivalent to (22).

(20) $[u_1\ u_2 \mid man\ u_1,\ woman\ u_2,\ u_1\ adores\ u_2,\ u_2\ abhors\ u_1]$

(21) $\lambda i\lambda j(i[u_1, u_2]j \wedge (man\ u_1)(j) \wedge (woman\ u_2)(j) \wedge (u_1\ adores\ u_2)(j) \wedge$
$(u_2\ abhors\ u_1)(j))$

(22) $\lambda i\lambda j(i[u_1, u_2]j \wedge man(\mathsf{V}(u_1)(j)) \wedge woman(\mathsf{V}(u_2)(j)) \wedge$
$adores(\mathsf{V}(u_1)(j))(\mathsf{V}(u_2)(j)) \wedge abhors(\mathsf{V}(u_2)(j))(\mathsf{V}(u_1)(j)))$

In a similar way all other boxes can be rewritten as certain terms $\lambda i\lambda j\varphi$, where φ is a first-order formula. Of course, for practical purposes we greatly prefer the more transparent box notation and in fact it will turn out to be completely unneccessary to expand definitions.

However, if we want to talk about the *truth* of a given sentence (given an input state) the box notation is no longer adequate and it seems better to switch to a more conventional predicate logical form in that case. We say that a condition γ is *true* in some state i in M if $\gamma(i)$ holds in M and that γ is *true* in M simpliciter if γ is true in all states i in M. We define a box K to be *true* in some state i if $\exists j K(i)(j)$ is true; K is *true* if K is true in all states i in M. This corresponds to the definition of truth in DRT as the existence of a verifying embedding. For example, the truth conditions of (22) can be rendered as (23).

(23) $\exists j(i[u_1, u_2]j \wedge man(\mathsf{V}(u_1)(j)) \wedge woman(\mathsf{V}(u_2)(j)) \wedge$
$adores(\mathsf{V}(u_1)(j))(\mathsf{V}(u_2)(j)) \wedge abhors(\mathsf{V}(u_2)(j))(\mathsf{V}(u_1)(j)))$

This, of course, is not the formula that we would normally associate with the truth-conditions of (1), the text which was represented by (20). But it is equivalent to it: using the Unselective Binding Lemma we readily reduce the unneccessarily complicated (23) to the more familiar (24) below.

(24) $\exists x_1 x_2\, (man(x_1) \wedge woman(x_2) \wedge adores(x_1)(x_2) \wedge abhors(x_2)(x_1))$

For a general algorithmic method for going from boxes to the predicate logical terms giving their truth-conditions, see III.5.

Replacing SEM1-SEM4 by ABB1-ABB4 meant a move from meta-language to object-language.[7] The gain is that we now have lambda-abstraction and application in our logic: it is now legal to use terms such as $\lambda v[\,|\, farmer\ v]$, where v is a type π variable, and $\lambda P'\lambda P([u_2 \,|\,]\ ;\ P'(u_2)\ ;\ P(u_2))$, where P and P' are variables of type $\pi(s(st))$. The first of these expressions may be considered to be a translation of the common noun `farmer` and the second a translation of the indexed indefinite determiner `a`^{2}. Since we are working in a standard type logic, λ-conversion holds, and it is easily seen that the application of the second term to the first is equivalent to $\lambda P([u_2 \,|\,]\ ;\ [\,|\, farmer\ u_2]\ ;\ P(u_2))$. Since the Merging Lemma is still valid under the new interpretation of the DRT constructs, as the reader may want to verify, the latter term further reduces to $\lambda P([u_2 \,|\, farmer\ u_2]\ ;\ P(u_2))$, which will be our translation of `a`^{2} `farmer`. This means that we can compute the meaning of this indefinite noun phrase from the meanings of its parts. In III.4 below we shall see how the method enables us to get a compositional translation of simple texts into boxes, but first we shall set up some syntactic scaffolding.

---

[7] Note that we have injected that part of the metatheory that deals with the assignment of values to variables into the object language. Compare this to what happens in modal logic, where models are treated as possible worlds and the consequence relation is treated as strict implication on the object-level (see Scott [1971]).

III.3 A Small Fragment of English: Syntax

In order to be able to illustrate how our compositional semantics works, we need to have the syntax of a small fragment of English at our disposal. The exact choice of formalism is quite unimportant here and we can largely remain agnostic about matters syntactic except inasfar as they effect the interpretability of the input to the semantic component. The most important requirement that we impose is that the syntactic component of grammar assigns indices to all names, pronouns and determiners. We shall follow the convention in Barwise [1987] and index antecedents with superscripts and dependent elements with subscripts. A pair consisting of a subscript n and that same number n occurring as a superscript may be thought of as an arrow going from the subscripted to the superscripted element. The indexed little text in (25), for example, should really be thought of as (26), where the anaphoric dependencies are shown with the help of arrows.

(25) $\texttt{A}^1$ `farmer owns` $\texttt{a}^2$ `donkey.` $\texttt{He}_1$ `beats` $\texttt{it}_2$.

(26) `A farmer owns a donkey. He beats it.`

We require that no two superscripts are the same, reflecting the fact that dependent elements can have only one antecedent and, since we are primarily interested in the phenomenon of linguistic anaphora as opposed to deixis, we demand that for each subscript n there is a superscript n, i.e. each dependent element has an antecedent. The relation between dependent elements and their antecedents thus becomes a total function, which we shall denote as *ant*. In (25), for example, *ant*($\texttt{it}_2$) will be $\texttt{a}^2$. There will be no further requirements on indexing as we want our semantic theory to predict which anaphoric relations can obtain, as far as semantics is concerned, and we do not want to stipulate matters a priori.

The syntax of our fragment will be couched in a transformational setting, a reduced version of the Government and Binding approach in the tradition of Chomsky [1981][8] (the reader will have no difficulty to transpose things to her favourite syntactical formalism). We assume that there are four components of syntax called D-Structure (DS), S-Structure (SS), Logical Form (LF) and Phonological Form (PF) respectively. The first three of these components will be defined as sets of labelled bracketings, or trees, connected by movement rules ('move α'). In our limited set-up we shall assume that S-Structure can be defined from D-Structure with the help of a rule called

[8] For a connection between traditional Montague Semantics and the full GB theory which is close to the present interpretation of a reduced GB theory on the basis of CDRT, see Bittner [1994]. For a connection between the present semantic theory and Lambek Categorial Grammar see Muskens [1994].

Relativization and that LF can be defined from SS using a rule called Quantifier Raising. PF will also be defined from SS and the overall set-up of the syntax of our fragment will thus conform to the standard picture below.

(27)
$$\begin{array}{ccc} & \text{DS} & \\ & | & \\ & \text{SS} & \\ \swarrow & & \searrow \\ \text{PF} & & \text{LF} \end{array}$$

Let us fill in some details. We define the DS component of our grammar to consist of all labelled bracketings that can be generated in the usual way with the help of phrase structure rules PS1–PS12 and lexical insertion rules LI1–LI9.

(PS 1)	T	$\rightarrow$	T S	(PS 7)	VP	$\rightarrow$	V′
(PS 2)	T	$\rightarrow$	S	(PS 8)	V′	$\rightarrow$	V_t NP
(PS 3)	S	$\rightarrow$	S′ S	(PS 9)	V′	$\rightarrow$	V_{in}
(PS 4)	S′	$\rightarrow$	IMP S	(PS 10)	NP	$\rightarrow$	DET N′
(PS 5)	S	$\rightarrow$	NP VP	(PS 11)	N′	$\rightarrow$	N
(PS 6)	VP	$\rightarrow$	AUX V′	(PS 12)	N′	$\rightarrow$	N S

(LI 1)	DET	$\rightarrow$	$\texttt{a}^n$, $\texttt{every}^n$, $\texttt{no}^n$, $\texttt{some}^n$	(LI 5)	N	$\rightarrow$	`farmer, boy`…
				(LI 6)	AUX	$\rightarrow$	`doesn't`, `don't`
(LI 2)	NP	$\rightarrow$	$\texttt{he}_n$, $\texttt{she}_n$, $\texttt{it}_n$	(LI 7)	V_t	$\rightarrow$	`own`,…
(LI 3)	NP	$\rightarrow$	`Mary`,…	(LI 8)	V_{in}	$\rightarrow$	`stink`,…
(LI 4)	NP	$\rightarrow$	`who`, `whom`, `which`	(LI 9)	IMP	$\rightarrow$	`if`

This part of the grammar should clearly be augmented with some *feature system* in order to rule out the worst overgeneration, but since it seems possible to handle the semantics of this particular fragment without taking recourse to interpreting features, the addition of such a system has been suppressed. Here are some examples of labelled bracketings (or, trees) that are elements of DS.

(28) $[_T[_T[_S[_{NP}[_{DET}\texttt{a}^1][_{N'}[_N\texttt{man}]]][_{VP}[_{V'}[_{V_t}\texttt{adores}][_{NP}[_{DET}\texttt{a}^2]$
$[_{N'}[_N\texttt{woman}]]]]]][_S[_{NP}\texttt{she}_2][_{VP}[_{V'}[_{V_t}\texttt{abhors}][_{NP}\texttt{him}_1]]]]]]$

(29) $[_{NP}[_{DET}\texttt{a}^7][_{N'}[_N\texttt{girl}][_S[_{NP}[_{DET}\texttt{every}^3][_{N'}[_N\texttt{boy}]]]$
$[_{VP}[_{V'}[_{V_t}\texttt{adores}][_{NP}\texttt{Mary}^4]]]]]$

(30) $[_{NP}[_{DET}\texttt{a}^7][_{N'}[_N\texttt{girl}][_S[_{NP}[_{DET}\texttt{every}^3][_{N'}[_N\texttt{boy}]]]$
$[_{VP}[_{V'}[_{V_t}\texttt{adores}][_{NP}\texttt{whom}]]]]]$

Clearly, structures like (29) are not grammatical. We could try to rule them out on the level of syntax, but have chosen to account for their ungrammaticality by not assigning them any meaning. Why (29) is not interpretable will become clear in the next section.

We now come to the definition of S-Structure. In order to connect D-Structures like (30) to acceptable forms of English that will be interpretable in our semantics, we shall employ a rudimentary version of the movement rule of *Relativization*. Informally speaking, Relativization allows us to move any relative pronoun from the position were it was generated and adjoin it to some S node higher up in the tree. The relative pronoun leaves a trace with which it becomes coindexed. More formally, we say that a tree (labelled bracketing) Ξ *follows by Relativization* from a tree Θ if and only if Ξ is the result of replacing some subtree of Θ of the form $[_S\, X\, [_{NP}\texttt{wh}]\, Y]$, where X and Y are (possibly empty) strings and `wh` is either `who`, `whom` or `which`, by a tree $[_S[_{NP}\texttt{wh}]^n[_S\, X\, e_n\, Y]]$, where n is any number not occurring in Θ. Abbreviated:

(Relativization) $\quad [_S\, X\, [_{NP}\texttt{wh}]\, Y] \;\Rightarrow\; [_S[_{NP}\texttt{wh}]^n[_S\, X\, e_n\, Y]]$

As an example note that (31) belows follow by Relativization from (30).

(31) $[_{NP}[_{DET}\texttt{a}^7][_{N'}[_N\texttt{girl}][_S[_{NP}\texttt{whom}]^6[_S[_{NP}[_{DET}\texttt{every}^3][_{N'}[_N\texttt{boy}]]]$
$[_{VP}[_{V'}[_{V_t}\texttt{adores}]e_6]]]]]$

The formal definition of S-Structure is that it is the smallest set of trees that includes D-Structure and is closed under Relativization. Thus DS $\subseteq$ SS, which may be a surprising definition in view of the fact that trees like (30) are not normally accepted as S-Structures. But again there is the question of interpretability: (30), unlike (31), will not receive an interpretation in the semantics to be defined below and in effect no harm is done.

We shall not attempt to give a serious definition of PF, as phonology is not a concern of this paper, but as a stop-gap we shall reduce phonology to orthography and stipulate that Phonological Forms are defined from S-Structures by the rule that any S-Structure without its brackets, indexes and traces is a Phonological Form. Thus (32) below is the Phonological Form that results from (31).

(32) `a girl whom every boy adores`

Lastly, we come to the definition of LF, the component of grammar that will be interpreted by our semantics. The motivation for having this component in syntax is—it must be confessed—entirely semantical in nature: on the one hand we want the language that is to be interpreted to be unambiguous (a 'disambiguated language' in Montague's sense), on the other we want to account for scope ambiguities like the one that arguably exists in (33).

(33) `every girl adores a boy`

The two different readings that this sentence seems to have will be obtained by adopting the rule of *Quantifier Raising* (May [1977]). Quantifier Raising adjoins NPs to S. Formally, a tree Ξ *follows by Quantifier Raising* from a tree Θ if and only if Ξ is the result of replacing some subtree Σ of Θ of the form $[_S\, X\, [_{NP}\, Z]\, Y]$, by $[_S[_{NP}\, Z]^n[_S\, X\, e_n\, Y]]$, where n is any number not occurring in Θ. Conditions on this rule are that $[_{NP}\, Z]$ is a tree, that Z is not not a (relative) pronoun, and that $[_{NP}\, Z]$ does not occur as a proper subtree of a subtree of Σ of the form $[_{NP}\, Z']$. We abbreviate the rule as follows

(Quantifier Raising) $[_S\, X\, [_{NP}\, Z]\, Y] \Rightarrow [_S[_{NP}\, Z]^n[_S\, X\, e_n\, Y]]$

In order to give some examples of applications of this rule we note that (35) follows by Quantifier Raising from the S-structure (34); (36) and (37) in their turn follow by Quantifier Raising from (35).[9]

(34) $[_S[_{NP}\texttt{every}^1\ \texttt{girl}]\ \texttt{adores}\ [_{NP}\texttt{a}^2\ \texttt{boy}]]$
(35) $[_S[_{NP}\texttt{a}^2\ \texttt{boy}]^3[_S[_{NP}\texttt{every}^1\ \texttt{girl}]\ \texttt{adores}\ e_3]]$
(36) $[_S[_{NP}\texttt{every}^1\ \texttt{girl}]^4[_S[_{NP}\texttt{a}^2\ \texttt{boy}]^3[_S e_4\ \texttt{adores}\ e_3]]]$
(37) $[_S[_{NP}\texttt{a}^2\ \texttt{boy}]^3[_S[_{NP}\texttt{every}^1\ \texttt{girl}]^4[_S e_4\ \texttt{adores}\ e_3]]]$

We define LF to be the smallest set of trees that contains SS and is closed under Quantifier Raising. Note that SS $\subseteq$ LF, which means that our rule of Quantifier Raising is optional, just like Relativization was. But while structures like (30) simply do not get an interpretation, structures like (34) and (35) do. The first will in fact be connected to the $\forall\exists$ reading of sentence (33) and the second to its $\exists\forall$ reading. Structures (36) and (37) will also be connected to these readings and so one might want to define only these to belong to LF proper, stipulating perhaps that every non-pronominal NP must be raised exactly once. But since this would not alter the set of meanings that are connected with any S-structure and since the present definition is simpler than the one that we would need in that case, we shall deviate from tradition and let the definition of LF be as it stands.

III.4. A Small Fragment of English: Semantics

Before we assign meanings to the LF expressions of the little fragment given in the preceding section, let us agree on some notational conventions. In Table 2 below some

---

[9] Note that, as an anonymous referee kindly pointed out, trees such as (36) and (37) can be the input of Quantifier Raising again, with results such as $[_S[_{NP}\texttt{every}^1\ \texttt{girl}]^4[_S e_4[_S[_{NP}\texttt{a}^2\ \texttt{boy}]^3[_S e_4\ \texttt{adores}\ e_3]]]]$. However, such structures will receive no translations by the rules that will be formulated in the next section and are therefore out on semantic grounds. We may, but need not, stipulate that such structures are out syntactically as well.

types are listed, together with the typography that we shall typically use when referring to variables and constants of these types. Since all logical expressions that we shall assign to expressions of the fragment will be equivalent to terms consisting of a chain of lambdas followed by an expression of the box type $s(st)$, it is useful to have a special notation for the types of such expressions: we shall abbreviate any type of the form $\alpha_1(\ldots(\alpha_n(s(st))\ldots)$ as $[\alpha_1\ldots\alpha_n]$. So boxes have type [] and, as will become clear below, we associate terms of type $[\pi]$ with common nouns and intransitive verbs, terms of type $[[[\pi]]\pi]$ with transitive verbs, terms of type $[[\pi]]$ with noun phrases, terms of type $[[\pi][\pi]]$ with determiners, etcetera. Verb phrases will either get $[[[\pi]]]$ or the more basic $[\pi]$, depending on whether they dominate an auxiliary or not.

Type	**Abbreviation**	**Name of objects**	**Variables**	**Constants**
et	—	static one-place predicates	—	*stinks, farmer,...*
$e(et)$	—	static two-place relations	—	*loves, owns,...*
$s(st)$	[]	dynamic propositions	p, q	—
$\pi(s(st))$	$[\pi]$	dynamic one-place predicates	P	—
$(\pi(s(st)))(s(st))$	$[[\pi]]$	dynamic one-place quantifiers	Q	—

Table 2.

Translations of lexical elements and indexed traces are given in rule T_0 below.[10] Note that each possible antecedent A (a determiner or a proper name) introduces a new discourse referent, which we shall denote as $dr(A)$. So, for example, $dr(\texttt{no}^{27})$ will be u_{27} and $dr(\texttt{John}^3)$ will be *John*. Anaphoric pronouns pick up the discourse referent that was created by their antecedent. In (38), for example, we have that $ant(\texttt{it}_8) = \texttt{a}^8$, and so $dr(ant(\texttt{it}_8)) = u_8$, which means that T_0 assigns $\lambda P(P(u_8))$ (the 'lift' of u_8) to $\texttt{it}_8$. Similarly, since $dr(ant(\texttt{she}_3)) = Sue$, the pronoun $\texttt{she}_3$ will be translated as $\lambda P(P(Sue))$.

(38) $\texttt{Sue}^3$ `has` $\texttt{a}^8$ `pigeon.` $\texttt{She}_3$ `feeds` $\texttt{it}_8$.

10 Not all lexical items are actually given here; but the ones that are absent should be treated in analogy with the nearest ones that are present, so e.g. `boy` translates as $\lambda v[\ |\ boy\ v]$ since the translation of `farmer` is given as $\lambda v[\ |\ farmer\ v]$.

T_0 BASIC TRANSLATIONS[11]

			type
`a`^{n}	$\rightsquigarrow$	$\lambda P'\lambda P([u_n \mid\,] ; P'(u_n) ; P(u_n))$	$[[\pi][\pi]]$
`no`^{n}	$\rightsquigarrow$	$\lambda P'\lambda P[\mid \mathbf{not}([u_n \mid\,] ; P'(u_n) ; P(u_n))]$	$[[\pi][\pi]]$
`every`^{n}	$\rightsquigarrow$	$\lambda P'\lambda P[\mid ([u_n \mid\,] ; P'(u_n)) \Rightarrow P(u_n)]$	$[[\pi][\pi]]$
`Mary`^{n}	$\rightsquigarrow$	$\lambda P.P(Mary)$	$[[\pi]]$
`he`$_n$	$\rightsquigarrow$	$\lambda P(P(\delta))$, where $\delta = dr(ant(\texttt{he}_n)$	$[[\pi]]$
e_n	$\rightsquigarrow$	$\lambda P(P(v_n))$	$[[\pi]]$
`who`	$\rightsquigarrow$	$\lambda P'\lambda P\lambda v(P(v) ; P'(v))$	$[[\pi][\pi]\pi]$
`farmer`	$\rightsquigarrow$	$\lambda v[\mid farmer\ v]$	$[\pi]$
`stink`	$\rightsquigarrow$	$\lambda v[\mid stinks\ v]$	$[\pi]$
`love`	$\rightsquigarrow$	$\lambda Q\lambda v(Q(\lambda v'[\mid v\ loves\ v']))$	$[[[\pi]]\pi]$
`doesn't`	$\rightsquigarrow$	$\lambda P\lambda Q[\mid \mathbf{not}\ Q(P)]$	$[[\pi][[\pi]]]$
`if`	$\rightsquigarrow$	$\lambda pq[\mid p \Rightarrow q]$	$[[][]]$

From these basic translations other terms translating complex expressions can be obtained by specifying how the translation of a mother node depends on the translations of its daughters. We provide five rules. The first says that single-daughter mothers simply inherit their translations from their daughters. The second allows for applying a translation of one daughter to a translation of another daughter and placing the result on the mother. The third is the sequencing rule that we have met in II.2.: a text followed by a sentence can be translated as a translation of that text sequenced with a translation of that sentence. The fourth rule handles quantifying-in. The last rule allows us to simplify translations. We say that a term β *follows by reduction* from a term α if we can obtain β from α by a finite number of lambda conversions and mergings, i.e. replacements of subterms $[u_1 \ldots u_n \mid \gamma_1,\ldots,\gamma_m] ; [u'_1 \ldots u'_k \mid \delta_1,\ldots,\delta_q]$ with $[u_1 \ldots u_n u'_1 \ldots u'_k \mid \gamma_1,\ldots,\gamma_m, \delta_1,\ldots,\delta_q]$, provided none of $u'_1 \ldots u'_k$ occurs in any of $\gamma_1,\ldots,\gamma_m$.[12] T_5 states that such reductions can be performed.

T_1 COPYING

If A $\rightsquigarrow$ α and A is the only daughter of B then B $\rightsquigarrow$ α.

---

[11] The category-to-type rule that has been at the back of my mind while writing down these translations is TYP(e) = π, TYP(t) = [] and TYP(A/B) = (TYP(B),TYP(A)), where e and t are basic categories and other categories are defined as in Montague [1973]. With the help of this category-to-type rule it is a relatively simple task to assign the present semantics to Generalised Categorial Grammars such as the Lambek Calculus (see Muskens [1994]). We could also connect the semantics with a subsystem of the *undirected* Lambek calculus to obtain a scoping and type-shifting system, as it is done in Hendriks [1993], who extends the rules of Partee & Rooth [1983].

[12] The provision can be relaxed somewhat by requiring that none of $u'_1 \ldots u'_k$ occurs *free* in any of $\gamma_1,\ldots,\gamma_m$. For the notion of a free occurrence of an unspecific referent see III.5.

T_2 APPLICATION
If A $\rightsquigarrow$ α, B $\rightsquigarrow$ β and A and B are the only daughters of C, then C $\rightsquigarrow$ $\alpha(\beta)$, provided that this is a well-formed term.

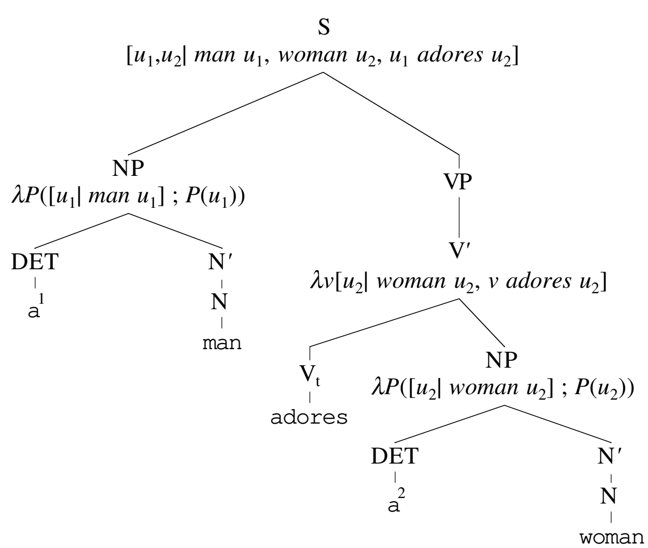
T_3 SEQUENCING
If T $\rightsquigarrow$ τ, S $\rightsquigarrow$ σ and T and S are daughters of T′ then T′ $\rightsquigarrow$ τ ; σ, provided that this is a well-formed term.

T_4 QUANTIFYING-IN
If NP^n $\rightsquigarrow$ η, S $\rightsquigarrow$ σ and NP^n and S are daughters of S′ then S′ $\rightsquigarrow$ $\eta(\lambda v_n \sigma)$, provided that this is a well-formed term.

T_5 REDUCTION
If A $\rightsquigarrow$ α and β follows from α by reduction then A $\rightsquigarrow$ β.

In these rules we identify trees with their top nodes. Formally, we define *translates as* to be the smallest relation $\rightsquigarrow$ between trees and logical expressions that conforms to T_0-T_5. It is useful to think of T_0-T_5 as rules for decorating trees. For example, (39) shows a decoration for the tree for $\texttt{a}^1$ `man adores` $\texttt{a}^2$ `woman` (only translations that are not simply copied from the lexicon T_0 or from a single daughter node are displayed).

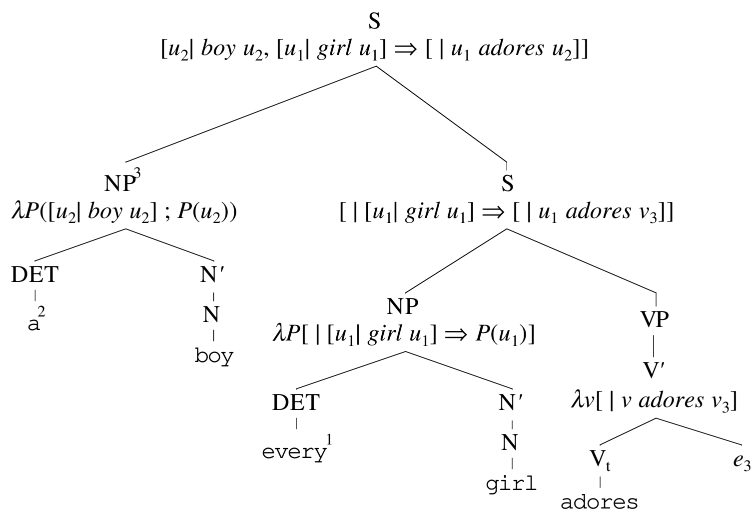
(39)



Here, since $\texttt{a}^2$ translates as $\lambda P'\lambda P([u_2 \mid\,] ; P'(u_2) ; P(u_2))$ and `woman` translates as $\lambda v[\,\mid$ *woman* $v]$, the tree for $\texttt{a}^2$ `woman` translates as the former term applied to the latter, which reduces to $\lambda P([u_2 \mid$ *woman* $u_2] ; P(u_2))$ with the help of lambda conversion and merging. The application of the translation of `adores` to this term then reduces to $\lambda v[u_2 \mid woman\ u_2,\ v\ adores\ u_2]$. Continuing the process in this way will lead to a decoration of the S node with the desired translation.

In order to give an example of an application of the Quantifying-in rule, I have decorated (35) with translations in (40) below. The trace e_3 introduces a free variable v_3 into the translation of the lower S node here, but, since the NP that is quantified-in is coindexed with this trace, the variable gets bound in the translation of the next node. The result is a translation that gives the indefinite $\texttt{a}^2$ `boy` scope over the quantified noun phrase $\texttt{every}^1$ `girl`. The discourse referent corresponding to the indefinite lands at the top of the box and can hence be picked up by pronouns later in the text. In (41) we see that such continuations are indeed possible. Of course, in the simple `every girl adores a boy` case strong preference is given to the narrow scope reading of `a boy`. But if we add a relative clause to the noun, for example, the wide scope interpretation becomes completely natural.

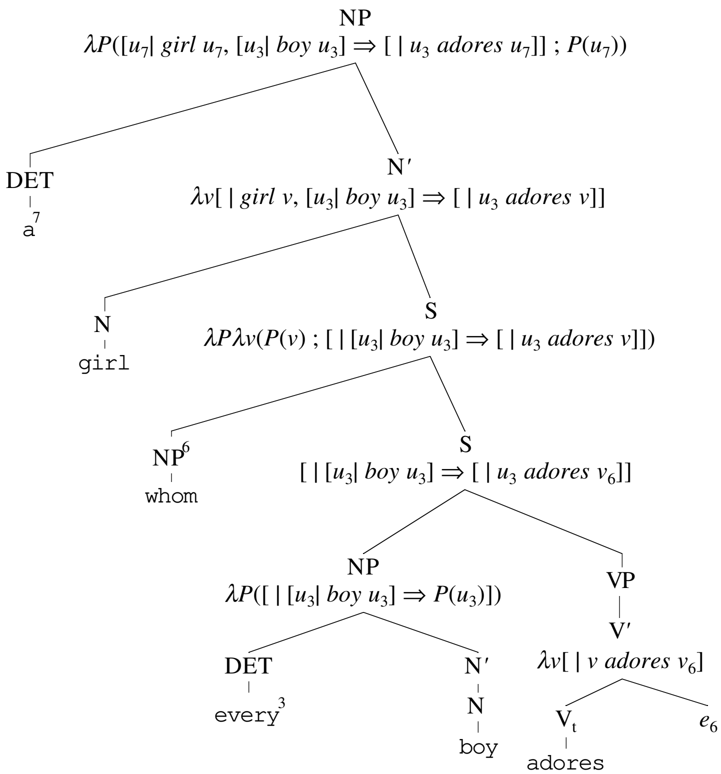
(40)



(41) Every girl adores a boy (I met yesterday). He adores only himself.

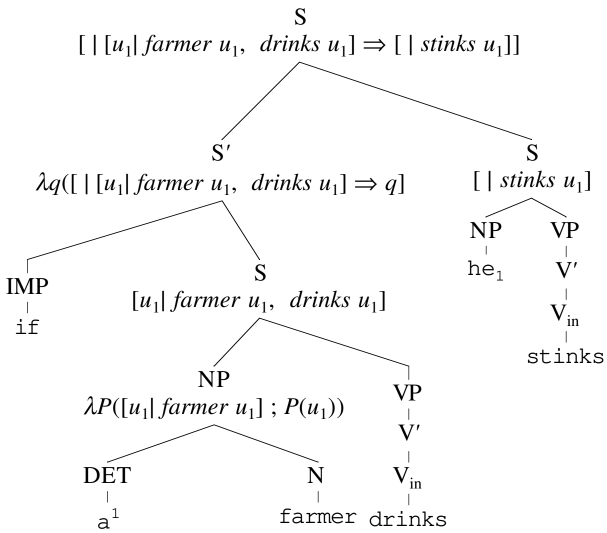
Structures that result from Quantifier Raising are not the only ones that allow for an application of the Quantifying-in rule. The results of Relativization can also be decorated with its help. In (42) an example is shown. The translation of `whom` is applied to $\lambda v_6[\ \mid [u_3 \mid boy\ u_3] \Rightarrow [\ \mid u_3\ adores\ v_6]]$ here, resulting in a term that is applicable to the translation of `girl`. Clearly, the position of the relative pronoun plays an essential role for the interpretability of the tree in this example. In (30), where the relative pronoun has not moved from its original position, the translation of `adores` cannot apply to the translation of `whom` and neither can `whom` apply to `adores`. Therefore the V′ and the rest of the tree above it will get no translation (note that our definition of the translation relation allows for this). In (29) the trouble arises when we want to apply the translation of `girl` to the translation of the S or vice versa. The same mechanism also renders those trees uninterpretable that result from two or more applications of Relativization to the same relative pronoun.

(42)



In Montague's PTQ (Montague [1973]) the Quantifying-in rules served two purposes: a) to obtain scope ambiguities between noun phrases and other scope bearing elements, such as other noun phrases, negations and intensional contexts, and b) to bind pronouns appearing in the expression that the noun phrase took scope over. In the present set-up the mechanism of discourse referents takes over the second task. In (43), for example, we see that the pronoun `he` can pick up the discourse referent that was introduced by `a farmer`, but no use has been made of Quantifier Raising or Quantifying-in.

(43)



III.5. Accessibility

A concept which plays an important role in the standard DRT construction algorithm is that of *accessibility*. Resolving a pronoun as an unspecific discourse referent u will result in a new occurrence of u in the discourse representation, but the pronoun cannot so be resolved unless an earlier occurrence of u is accessible from this new occurrence. Accessibility will play an important role here too and, for any given DRS K and any given occurrence of a variable u in an atomic condition in K, we shall define the set **acc**(u, K) of discourse referents that are accessible from u in K. In order to be able to define this set inductively, we are interested in sets **acc**(u, γ), where γ is a condition, as well. As a preliminary step we let the set of *active discourse referents,* **adr**(K), of any box K be defined by:

$\mathbf{adr}([u_1,\ldots,u_n \mid \gamma_1,\ldots,\gamma_m]) = \{u_1,\ldots,u_n\}$
$\mathbf{adr}(K_1 \,;\, K_2) = \mathbf{adr}(K_1) \cup \mathbf{adr}(K_2)$

Intuitively, the active discourse referents of K are those referents that K can 'bind to its right'; for example the discourse referents that an antecedent of a conditional can bind in the consequent. The following rules define the function **acc**.

(i)	$\mathbf{acc}(u, \varphi)$	=	$\varnothing$, if φ is atomic
(ii)	$\mathbf{acc}(u, \mathbf{not}\ K)$	=	$\mathbf{acc}(u, K)$
(iii)	$\mathbf{acc}(u, K_1\ \mathbf{or}\ K_2)$	=	$\mathbf{acc}(u, K_1)$, if u occurs in K_1
		=	$\mathbf{acc}(u, K_2)$, if u occurs in K_2
(iv)	$\mathbf{acc}(u, K_1 \Rightarrow K_2)$	=	$\mathbf{acc}(u, K_1)$, if u occurs in K_1
		=	$\mathbf{acc}(u, K_2) \cup \mathbf{adr}(K_1)$, if u occurs in K_2
(v)	$\mathbf{acc}(u, [u_1,\ldots,u_n \mid \gamma_1,\ldots,\gamma_m])$	=	$\mathbf{acc}(u, \gamma_i) \cup \{u_1,\ldots,u_n\}$, if u occurs in γ_i

(vi) $\mathbf{acc}(u, K_1 ; K_2)$ $= \mathbf{acc}(u, K_1)$, if u occurs in K_1
$= \mathbf{acc}(u, K_2) \cup \mathbf{adr}(K_1)$, if u occurs in K_2

We say that u' is *accessible* from an occurrence of u in K if $u' \in \mathbf{acc}(u, K)$; an occurrence of an unspecific referent u in an atomic condition is *free* in K if $u \notin \mathbf{acc}(u, K)$ and u is *free* in γ if $u \notin \mathbf{acc}(u, \gamma)$. A DRS that contains no free referents is a *proper* DRS. In III.3. we have imposed a very liberal indexing mechanism requiring only that the relation *ant* between dependent elements and their antecedents be a total function from subscripts to superscripts. We now distinguish between those indexed texts which are semantically acceptable and those which are not by requiring the former to have a proper DRS as a translation.

The notion of being a proper DRS is *representational* in the sense that a proper box K and a box K' which is not proper may have exactly the same semantic value. For example, (45), the translation of (44), in any model denotes exactly the same relation between states as (47), the translation of (46). Yet (45) is a proper DRS while (47) is not.

(44) No^1 `girl walks`
(45) $[\mid \mathbf{not}[u_1 \mid \mathit{girl}\ u_1, \mathit{walk}\ u_1]]$

(46) *No^1 `girl walks. If she`$_1$ `talks she`$_1$ `talks`
(47) $[\mid \mathbf{not}[u_1 \mid \mathit{girl}\ u_1, \mathit{walk}\ u_1], [\mid \mathit{talk}\ u_1] \Rightarrow [\mid \mathit{talk}\ u_1]]$

That a proper box and a box which is not proper can have the same semantic value is the reason why we have not allowed full closure under equivalence in rule T_5. If a node of a tree is decorated with a certain translation we may with safety perform lambda conversions and merging. But if we would make it a general rule that terms can be replaced by their equivalents, proper DRSs would be replaced by ones that are not proper and vice versa and the characterisation of acceptable indexings would be lost.

A semantic theory is *non-representational* if it accounts for semantic phenomena in terms of semantic values only. Although a non-representational theory may work with representations intermediate between language and semantic values, these should in principle be eliminable, just as the expressions of IL were eliminable in principle from Montague's set-up of the PTQ fragment. But many accounts of anaphora which present themselves as non-representational, including the theories of Groenendijk & Stokhof [1990, 1991] and Muskens [1991, 1995a], have a problem with (44) and (46). As these theories assign the same semantic values to (44) and (46), they wrongly predict (46) to be as acceptable as (44) is.[13]

---

[13] DPL, for example, assigns $\neg\exists x_1(\mathit{girl}\ x_1 \wedge \mathit{walk}\ x_1)$ to (44), while it assigns $\neg\exists x_1(\mathit{girl}\ x_1 \wedge \mathit{walk}\ x_1) \wedge (\mathit{talk}\ x_1 \rightarrow \mathit{talk}\ x_1)$ to (46). But these representations are equivalent and therefore the two

III.6. Truth and entailment: a weakest precondition calculus

We have now in fact connected the sentences and texts of our fragment with truth-conditions. If we are presented with a text then, in order to find its truth-conditions, we may parse it (find a Logical Form for it) according to the grammar given in III.3. and then decorate the resulting tree according to the rules given in III.4. The truth-conditions that we are after will be those of the box K that we find at the top node. By definition K will be true in a state i if $\exists jK(i)(j)$ is true. As we shall see shortly, the choice of i is immaterial if K is a proper DRS: $\exists jK(i)(j)$ will be true for all i or none then.

In III.2 we have shown how the truth conditions of a box can be found by expanding definitions and by applying the Unselective Binding Lemma. For simple DRSs such as (3) this is a trivial task, but if we apply the method to more complex discourse representations, things soon become involved and we find that the advantages of our system of abbreviations are lost. For this reason it is useful to know that there is a simple calculus which, given some box or condition, always outputs its truth conditions in the form of a first order formula. The calculus is an adaptation of a technique that was developed in computer science (the field of 'Hoare logics') and was first used for linguistic purposes in Van Eijck & De Vries [1992]. We give a variant here, defining a one-place translation function **tr** which sends conditions to predicate logical formulae and a two-place function **wp** (*weakest precondition*), which gives a formula of predicate logic if we feed it a box and a first-order formula.[14, 15] The idea is that $\mathbf{tr}(\gamma)$ gives the truth conditions of γ and that the truth conditions of K are given by $\mathbf{wp}(K,\top)$ (where $\top$ is the sentence that is always true). Assume that the set of variable referents $\{u_1, u_2,\ldots,u_n,\ldots\}$ and the set of individual variables $\{x_1, x_2,\ldots,x_n,\ldots\}$ are ordered as indicated. Let † be the function from discourse referents to individual variables and constants such that $u_n{}^\dagger = x_n$ for any n and $Tom^\dagger = tom$, $Tim^\dagger = tim$, $Mary^\dagger = mary$, etc. The following clauses do the job.

$$(\mathrm{TR_A}) \qquad \mathbf{tr}(R\{\delta_1,\ldots,\delta_n\}) \qquad = \quad R(\delta_1{}^\dagger)\ldots(\delta_n{}^\dagger)$$

---

texts should be treated as equally acceptable. Saying that (46) is out because its representation contains a free variable, while (44) is in because its representation is closed, involves an appeal to properties of the representations that are not mirrored in their semantics. It seems probable that this feature of DPL can be repaired by following the usual set-up of DRT and using finite assignments instead of total assignments as the basis of the semantic theory. I expect that a similar move could also change the present theory into a non-representational one.

14 For an alternative translation of DRT into predicate logic see Kamp & Reyle [1993].

15 The reason that $\mathbf{wp}(K, \chi)$ is called the *weakest precondition* of K and χ is that it corresponds to the set of all states i such that K, when started in state i, may after execution end in a state where χ holds. A predicate logical formula ψ is called a *precondition* of K and χ if in any state where ψ holds we can successfully carry out K and end in a state where χ holds. By the definition of $\mathbf{wp}(K, \chi)$ it is a precondition of K and χ and since $\mathbf{wp}(K, \chi)$ follows from any precondition of K and χ, it can be reasonably called the *weakest* precondition of K and χ.

	$\mathbf{tr}(\delta_1\ \mathbf{is}\ \delta_2)$	=	$\delta_1{}^\dagger = \delta_2{}^\dagger$
($\mathrm{TR}_{\mathbf{not}}$)	$\mathbf{tr}(\mathbf{not}\ K)$	=	$\neg\mathbf{wp}(K, \top)$
($\mathrm{TR}_{\mathbf{or}}$)	$\mathbf{tr}(K_1\ \mathbf{or}\ K_2)$	=	$\mathbf{wp}(K_1, \top) \vee \mathbf{wp}(K_2, \top)$
($\mathrm{TR}_{\Rightarrow}$)	$\mathbf{tr}(K_1 \Rightarrow K_2)$	=	$\neg\mathbf{wp}(K_1, \neg\mathbf{wp}(K_2, \top))$
($\mathrm{WP}_{[]}$)	$\mathbf{wp}([u_{k_1},\ldots,u_{k_n} \mid \gamma_1,\ldots,\gamma_m], \chi)$	=	$\exists x_{k_1} \ldots x_{k_n} (\mathbf{tr}(\gamma_1) \wedge\ldots\wedge \mathbf{tr}(\gamma_m) \wedge \chi)$
($\mathrm{WP}_{;}$)	$\mathbf{wp}(K_1 ; K_2, \chi)$	=	$\mathbf{wp}(K_1, \mathbf{wp}(K_2, \chi))$

As an example of how this may be used to find the truth conditions of any DRS, we compute those of $K = [u_6\ u_2 \mid man\ u_6,\ woman\ u_2,\ u_6\ adores\ u_2,\ u_6\ abhors\ u_1]$ by first writing down $\mathbf{wp}(K, \top)$, then applying $\mathrm{WP}_{[]}$, and then applying TR_{A} four times:

$\mathbf{wp}([u_6\ u_2 \mid man\ u_6,\ woman\ u_2,\ u_6\ adores\ u_2,\ u_6\ abhors\ u_1], \top)$	
$\exists x_6 x_2(\mathbf{tr}(man\ u_6) \wedge \mathbf{tr}(woman\ u_2) \wedge \mathbf{tr}(u_6\ adores\ u_2) \wedge \mathbf{tr}(u_2\ abhors\ u_6))$	$\mathrm{WP}_{[]}$
$\exists x_6 x_2(man(x_6) \wedge woman(x_2) \wedge adores(x_6)(x_2) \wedge abhors(x_2)(x_6))$	$4\times\mathrm{TR}_{\mathrm{A}}$

The following derivation is a bit longer, as the box that it starts with is a bit more complex.

$\mathbf{wp}([\ \mid [u_1\ u_2 \mid man\ u_1,\ woman\ u_2,\ u_1\ bores\ u_2] \Rightarrow [\ \mid u_2\ ignores\ u_1]], \top)$	
$\mathbf{tr}([u_1\ u_2 \mid man\ u_1,\ woman\ u_2,\ u_1\ bores\ u_2] \Rightarrow [\ \mid u_2\ ignores\ u_1])$	$\mathrm{WP}_{[]}$
$\neg\mathbf{wp}([u_1\ u_2 \mid man\ u_1,\ woman\ u_2,\ u_1\ bores\ u_2], \neg\mathbf{wp}([\ \mid u_2\ ignores\ u_1], \top))$	$\mathrm{TR}_{\Rightarrow}$
$\neg\mathbf{wp}([u_1\ u_2 \mid man\ u_1,\ woman\ u_2,\ u_1\ bores\ u_2], \neg\mathbf{tr}(u_2\ ignores\ u_1))$	$\mathrm{WP}_{[]}$
$\neg\mathbf{wp}([u_1\ u_2 \mid man\ u_1,\ woman\ u_2,\ u_1\ bores\ u_2], \neg ignores(x_2)(x_1))$	TR_{A}
$\neg\exists x_1 x_2(\mathbf{tr}(man\ u_1) \wedge \mathbf{tr}(woman\ u_2) \wedge \mathbf{tr}(u_1\ bores\ u_2) \wedge \neg ignores(x_2)(x_1))$	$\mathrm{WP}_{[]}$
$\neg\exists x_1 x_2(man(x_1) \wedge woman(x_2) \wedge bores(x_1)(x_2) \wedge \neg ignores(x_2)(x_1))$	$3\times\mathrm{TR}_{\mathrm{A}}$

Clearly, applied to any box K, the method will give a result in a number of steps that is equal to the complexity of K, if we define the latter as the number of applications of ABB1-ABB4 that were needed to form the box.

It is worth noting that the notion of accessibility is closely related to the notion of weakest precondition. In particular, we find that the following relation obtains.

PROPOSITION 1. *A DRS K is proper if and only if* $\mathbf{wp}(K,\top)$ *is a closed formula*

To prove this statement we must consider a somewhat messier one, namely the conjunction of (a) u_k is free in γ iff x_k is free in $\mathbf{tr}(\gamma)$, and (b) x_k is free in $\mathbf{wp}(K, \chi)$ iff u_k is free in K, or x_k is free in χ while $u_k \notin \mathbf{adr}(K)$. This will follow by a straightforward induction on the complexity of constructs in the discourse representation language and entails the proposition.

The following proposition justifies the calculus in the sense that it shows that $\mathbf{wp}(K,\top)$ really gives the truth conditions of K, provided K is a proper DRS.

PROPOSITION 2. *If* $\mathbf{wp}(K, \top)$ *is closed then it is equivalent to* $\exists j K(i)(j)$ *for any i.*

The proposition is proved via a generalisation which also gives information about cases where $\mathbf{wp}(K,\top)$ is an open formula. The proof is given in an appendix.

The weakest precondition calculus helps characterise the relation of entailment on boxes. Define $K_1,\ldots,K_n \models_{DRT} K$ to hold just in case, for every M and i in M, if $K_1,\ldots,K_n$ are true in i in M, then K is true in i in M. We have:

PROPOSITION 3. $K_1,\ldots,K_n \models_{DRT} K$ iff $\mathbf{wp}(K_1,\top),\ldots,\mathbf{wp}(K_n,\top) \models \mathbf{wp}(K,\top)$

This time there is no restriction to closed formulae. For a short proof see the appendix again.

With proposition 3 we get a syntactic characterization of the entailment relation in DRT, via any of the familiar axiomatisations of first-order entailment.[16] A practical consequence is that a theorem prover for DRT can simply consist of an existing theorem prover for predicate logic plus seven additional Prolog clauses embodying the weakest precondition calculus.

An alternative notion of entailment (for DPL) was defined in Groenendijk & Stokhof [1991], whose definition boils down to the requirement that K follows from $K_1,\ldots,K_n$ if and only if $(K_1 ; \ldots ; K_n) \Rightarrow K$ is true in all models in all states. We denote this as $K_1,\ldots,K_n \models_{DPL} K$ and find as a direct consequence of Proposition 3:

COROLLARY. $K_1,\ldots,K_n \models_{DPL} K$ iff $\models \mathbf{tr}((K_1 ; \ldots ; K_n) \Rightarrow K)$

And again the question whether a given argument is valid under this notion of entailment can be reduced to a question about validity in elementary logic.

IV. AN ILLUSTRATION: ANAPHORA AND GENERALISED COORDINATION

By way of example I shall now apply the system to a field were the phenomena that DRT deals nicely with and those which are nicely treated in Montague Semantics seem to interact: anaphora across generalised coordinations. One of Kamp's main motivations for setting up the DRT framework was the paucity of Montague's treatment of anaphora. On the other hand, Kamp & Reyle [1993, pages 214-232] obviously have difficulty dealing with conjunctions and are forced to introduce considerable overall complications of their theory in order to cope with them. As

[16] Alternatively, we can characterise DRT entailment directly, using a natural deduction system as in Saurer [1993]. Saurer notices that DRT entailment can be characterised via translation into elementary logic, but claims psychological reality for his inference system, saying that it can be understood as offering a mental logic.

Montague's system is particularly apt at dealing with coordinating constructions, it seems that a treatment of anaphora, coordinations and their interplay which combines the good parts of both systems is called for. In the next pages I shall try to set up such a treatment.

It seems to be a fundamental fact about languages that expressions belonging to the same category can be coordinated. As long as A and B belong to the same category, A `and` B and A `or` B will belong to that category too. More in general, it seems that the following rules should be added to our phrase structure grammar.

(PS 13)	X	$\rightarrow$	X^+ CONJ X
(LI 10)	CONJ	$\rightarrow$	`and`, `or`

Here X may be unified with any category and X^+ stands for strings of Xs of any positive length. So the result of an application of PS 13 could be S CONJ S, or NP NP NP CONJ NP, as in `Tom, Dick, Harry and a girl who abhors Delaware`.

Now various authors (Von Stechow [1974], Keenan & Faltz [1978], Gazdar [1980], Partee & Rooth [1983]) have noted that the semantics of these coordinated constructions conforms to a striking regularity. Roughly speaking, as long as we are not considering cases where anaphoric dependencies play a role and barring 'group readings' such as in `Tom, Dick and Harry carried the piano`, the hypothesis that `and` always denotes intersection, and that `or` always denotes union seems to fare remarkably well.[17,18]

However, it seems that this Boolean hypothesis about coordination can no longer be upheld in a setting where anaphoric dependencies are taken into consideration. In (48) and (49) a minimal pair of short two sentence texts is given which shows that the pronoun $\texttt{it}_1$ can pick up a referent that was created by the indefinite determiner $\texttt{a}^1$ in a preceding sentence but that it cannot pick up the referent that was created by a determiner $\texttt{no}^1$ occurring in the same position (both standard DRT and our version

---

[17] In an extensional Montague semantics we could implement this Boolean hypothesis by always translating $A_1 \ldots A_n$ `and` A_{n+1} as (I) and $A_1 \ldots A_n$ `or` A_{n+1} as (II) below.

(I) $\lambda x_1 \ldots \lambda x_m(A_1'(x_1)\ldots(x_m) \wedge \ldots \wedge A_{n+1}'(x_1)\ldots(x_m))$
(II) $\lambda x_1 \ldots \lambda x_m(A_1'(x_1)\ldots(x_m) \vee \ldots \vee A_{n+1}'(x_1)\ldots(x_m))$

Here $A_1', \ldots, A_{n+1}'$ are assumed to be the translations of $A_1, \ldots, A_{n+1}$ respectively and $x_1, \ldots, x_m$ are variables of the types that are needed to make these well-formed terms.

[18] Of course the empirical content of the hypothesis depends on the types of the translations that we assign to expressions of different categories. There is also a characteristic flip-flop behaviour of the connectives: *Most men and women swim* means that most men swim and most women swim but *My friend and colleague swims* means that a person who is both my friend and my colleague swims. The word 'and' cannot be treated as intersection in both cases. This observation is attributed to Robin Cooper by Partee & Rooth [1983] who propose to deal with the phenomenon via type-shifting.

predict this, of course). The pair (50)-(51) shows that the same phenomenon occurs across sentence conjunctions. Since we have accepted that the sequencing of sentences cannot be treated as simple logical conjunction we must also accept that a grammatical conjunction of sentences cannot be treated in this way.

(48) `A`^{2} `cat catches a`^{1} `fish. It`$_2$ `eats it`$_1$.
(49) `*A`^{2} `cat catches no`^{1} `fish. It`$_2$ `eats it`$_1$.

(50) `A`^{2} `cat catches a`^{1} `fish and it`$_2$ `eats it`$_1$.
(51) `*A`^{2} `cat catches no`^{1} `fish and it`$_2$ `eats it`$_1$.

But the phenomenon is not restricted to conjunctions of sentences, of course. In (52)-(53) and in (54)-(55) it is shown that it also occurs in conjunctions of expressions in categories other than S. (52)-(53) illustrate the case for VP and (54)-(55) for NP.

(52) `A`^{2} `cat catches a`^{1} `fish and eats it`$_1$
(53) `*A`^{2} `cat catches no`^{1} `fish and eats it`$_1$

(54) `John`^{3} `has a`^{2} `cat which catches a`^{1} `fish and a`^{4} `cat which eats it`$_1$
(55) `*John`^{3} `has a`^{2} `cat which catches no`^{1} `fish and a`^{4} `cat which eats it`$_1$

Fortunately it is possible to generalise the Boolean hypothesis of coordination to make it account for data such as these. While the Boolean theory equates `and` with a generalised form of logical conjunction and `or` with a generalised logical disjunction, we can equate the first with a generalised form of sequencing (relational composition) and the second with a generalisation of the DRT notion of sentence disjunction. Formally, we add the following schemata to T_0.

`and` $\rightsquigarrow \lambda R_1 \ldots \lambda R_n \lambda X_1 \ldots \lambda X_m (R_1(X_1)\ldots(X_m) \,;\, \ldots \,;\, R_n(X_1)\ldots(X_m))$
`or` $\rightsquigarrow \lambda R_1 \ldots \lambda R_n \lambda X_1 \ldots \lambda X_m [\,|\, R_1(X_1)\ldots(X_m) \textbf{ or } \ldots \textbf{ or } R_n(X_1)\ldots(X_m)]$,

These clauses assign an infinity of systematically related translations to `and` and `or`. Note that in any well-formed term of one of the displayed forms, the variables $R_1,\ldots, R_n$ must all have the same type.

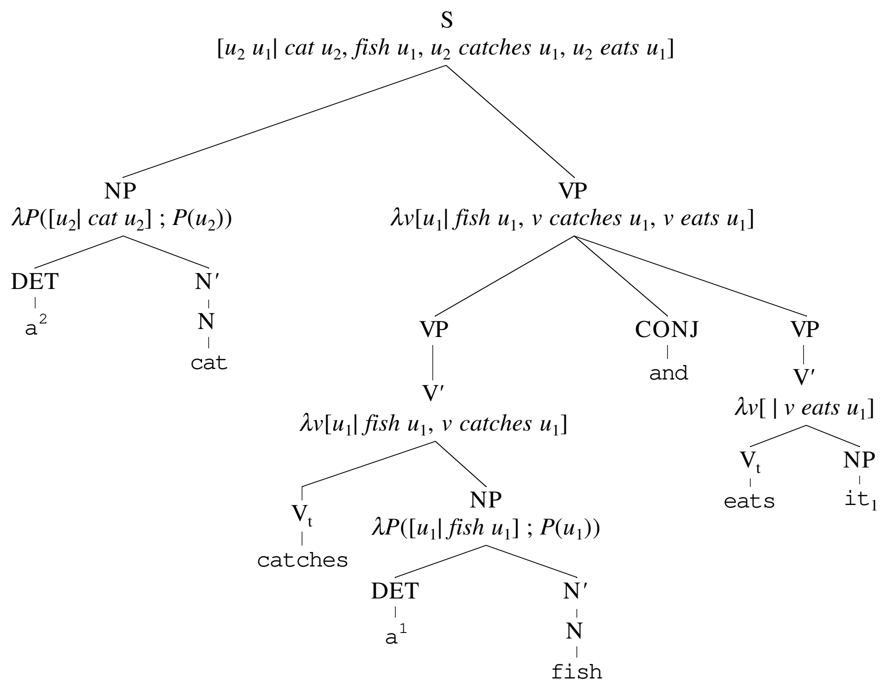
We now let the translation of a coordinated construction consist of the translation of the coordinating element applied to the translations of the coordinated expressions. More precisely, we add T_6 to our translation rules.

T6 GENERALISED COORDINATION
If $A_1 \rightsquigarrow \alpha_1, \ldots, A_{n+1} \rightsquigarrow \alpha_{n+1}$, CONJ$\rightsquigarrow \beta$, and $A_1, \ldots, A_n$, CONJ and A_{n+1} are daughters of A (in that order), then A $\rightsquigarrow \beta(\alpha_1)\ldots(\alpha_{n+1})$, provided $\beta(\alpha_1)\ldots(\alpha_{n+1})$ is well-formed and has the same type as each of the α_i.

It may be observed that only the 'right' translation of CONJ is applicable here.

In order to show how this works we give some examples. In (56) a tree for (52) is decorated with translations. Here only the translation $\lambda P'\lambda P\lambda v(P'(v) ; P(v))$ for `and` is applicable and the result of applying this to the translations of the coordinated VPs is shown on the mother VP.

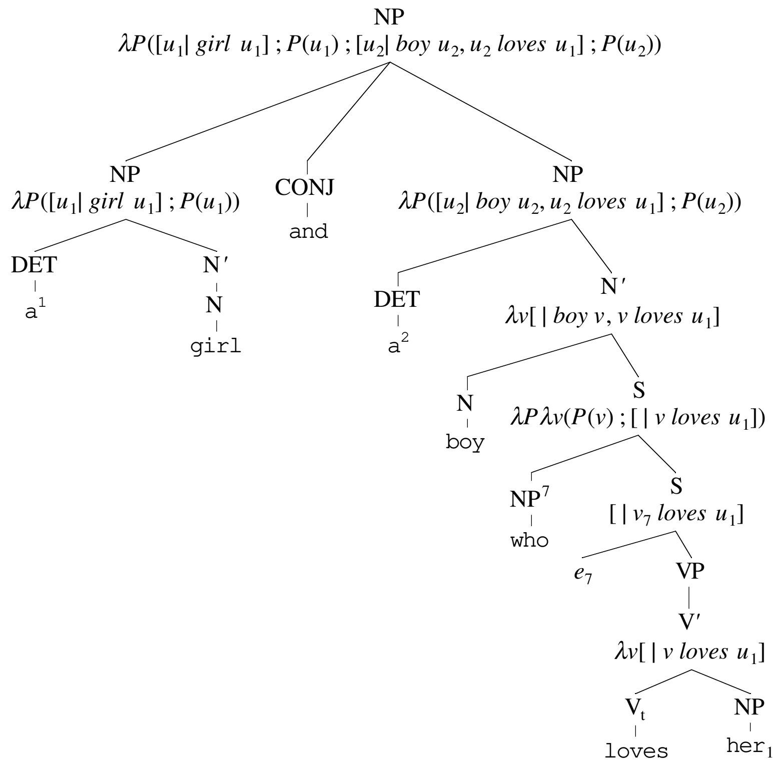
(56)



The reader may wish to verify that a similar tree for (53) leads to a box in which the discourse referent that is introduced by the word `no`[1] is not accessible from the discourse referent that is connected to the pronoun $\texttt{it}_1$.

A second example in (57) shows how the rule deals with coordinations of noun phrases. Here the translation $\lambda Q'\lambda Q\lambda P(Q'(P) ; Q(P))$ is the only one that can be used and its application to the two coordinated NPs leads to the translation at the top node. In (58) a full sentence with this NP in object position is given; in (59) its translation, and in (60) its truth conditions (obtained directly or via the weakest precondition calculus) are shown. Since (59) is a proper DRS (or, equivalently, since (60) is a closed formula) we find that (58) is acceptable.

(57)



(58) $\texttt{John}^3$ `admires` $\texttt{a}^1$ `girl and` $\texttt{a}^2$ `boy who loves` $\texttt{her}_1$

(59) $[u_1\ u_2 \mid girl\ u_1,\ John\ admires\ u_1,\ boy\ u_2,\ u_2\ loves\ u_1,\ John\ admires\ u_2]$

(60) $\exists x_1 x_2(girl(x_1) \wedge admires(john)(x_1)) \wedge boy(x_2) \wedge loves(x_2)(x_1) \wedge admires(john)(x_2))$

However, if we replace the determiner $\texttt{a}^1$ by $\texttt{no}^1$, as is done in (61), binding fails. Applying $\lambda Q'\lambda Q\lambda P(Q'(P) ; Q(P))$ to the translation of $\texttt{no}^1$ `girl`, $\lambda P[\mid \mathbf{not}([u_1|\ girl\ u_1] ; P(u_1))]$ and then to the translation of $\texttt{a}^2$ `boy who loves` $\texttt{her}_1$ now results in (62); application of the translation of `admires` to (62) results in (63); and a final application of the translation of $\texttt{John}^3$ gives (64), the translation of the sentence. We find that it contains a free occurrence of u_1. If we use the **wp** calculus to compute (64)'s truth conditions, we get the result in (65), an open formula with x_1 free. This means that (61) is predicted to be out: the quantifier $\texttt{no}^1$ `girl` does not manage to bind the dependent element $\texttt{her}_1$.

(61) *$\texttt{John}^3$ `admires` $\texttt{no}^1$ `girl and` $\texttt{a}^2$ `boy who loves` $\texttt{her}_1$

(62) $\lambda P([\mid \mathbf{not}([u_1|\ girl\ u_1] ; P(u_1))] ; [u_2 \mid\ boy\ u_2,\ u_2\ loves\ u_1] ; P(u_2))$

(63) $\lambda v([\mid \mathbf{not}([u_1|\ girl\ u_1] ; [\mid v\ admires\ u_1])] ; [u_2 \mid\ boy\ u_2,\ u_2\ loves\ u_1] ;$
$[\mid v\ admires\ u_2])$

(64) $[u_2 \mid \mathbf{not}[u_1|\ girl\ u_1,\ John\ admires\ u_1],\ boy\ u_2,\ u_2\ loves\ u_1,$

$John\ admires\ u_2$]

(65) $\exists x_2(\neg\exists x_1(girl(x_1) \wedge admires(john)(x_1)) \wedge boy(x_2) \wedge loves(x_2)(x_1) \wedge admires(john)(x_2))$

A last example[19] shows that our treatment of coordination sometimes leads to applications of the sequencing operator ';' which cannot be reduced with the help of the Merging Lemma as the conditions under which that rule is applicable are not fulfilled. In order to find the translation of the distributive reading of (66) we first conjoin the translations of `Bill`^{1} and `Sue`^{2} to $\lambda P(P(Bill)\ ;\ P(Sue))$. This term can then be applied to $\lambda v[u_3 \mid donkey\ u_3,\ v\ owns\ u_3]$, the translation of the VP. The result (after lambda conversions) is (67); but note that this extended DRS cannot be reduced with the help of merging, as the u_3 that is declared in the second box already appears in the conditions of the first box.

(66) `Bill`^{1} `and Sue`^{2} `own a`^{3} `donkey`

(67) $[u_3 \mid donkey\ u_3,\ Bill\ owns\ u_3]\ ;\ [u_3 \mid donkey\ u_3,\ Sue\ owns\ u_3]$

(68) $\exists x_3(donkey(x_3) \wedge owns(bill)(x_3)) \wedge \exists x_3(donkey(x_3) \wedge owns(sue)(x_3))$

This brings us outside the standard DRT language, but no harm is done as we systematically get the right predictions about truth conditions and anaphora in these cases. Using the program metaphor that is often useful when thinking about DRT we may say that the first box in (67) assigns a donkey which Bill owns to u_3 and that the second box then assigns it a possibly *new* value, a donkey owned by Sue. In other words, (67) is a case of *reassignment.* Clearly, first assigning a donkey owned by Bill to u_3 and then one owned by Sue is a program which can be carried out just in case Bill and Sue each own a donkey and indeed we find that our **wp** calculus correctly computes the truth conditions given in (68).

V. Conclusion and Further Prospects

In this paper we have defined a fusion of two important frameworks for natural language semantics, taking care that the resulting formalism is easy to work with and has a simple underlying mathematics. In order to obtain the constructs of DRT within the type logic which underlies Montague Semantics we have used a technique which I would like to call *grafting*: logics L and L' can often be combined by (a) taking an adequate Tarski definition of L', (b) axiomatising within L the concepts that this definition talks about, and then (c) introducing the syntactic constructs of L' as abbreviations within L by means of a transcription of the Tarski definition. L' can then

[19] Questions from Bart Geurts at the Ninth Amsterdam Colloquium and from an anonymous referee drew my attention to this example. I would like to thank them both.

be said to be *grafted upon L.* In our case L was many-sorted type logic, L' was DRT, the Tarski definition was the one given by Groenendijk & Stokhof [1991] and the transcription was given in ABB1-ABB4. But the technique seems very general, can be used whenever L' has an acceptable Tarski definition and L is expressive enough to transcribe this definition, and might help to integrate many different logics that have been proposed in order to deal with special phenomena in the semantics of natural language. There has been a tendency in the field to propose a different logic for each phenomenon in natural language, but ultimately language is one, and we shall need to synthesize these logics if we want to obtain one logic for the whole range of phenomena. Grafting logics upon a single expressive logic such as classical type logic is one possible technique to do this.

There are two directions that can be taken from here, and both must be explored. The first leads into linguistics and the second leads further into logic. Concerning the first: In this paper I have hardly used the formalism presented here to work out any new descriptive theories. The linguistic theory of section III (as opposed to the general *formalism* which was presented there) is of course nothing but a streamlined presentation of the theories by Kamp and Heim. The work in section IV is an improvement over the treatment in Kamp & Reyle [1993] as far as I can see, but not because it describes essentially new phenomena, or because it can deal with phenomena that Kamp & Reyle cannot treat, but because it can deal with the data in a principled way, without having to complicate the theory, and assigning a single algebraic operation to each of the words `and` and `or`, while Kamp & Reyle's treatment introduces complexities that threaten the DRT framework as such. In this paper my main purpose has been to define a practicable formalism for the analysis of natural language semantics, but the next step must of course consist of a series of applications of the formalism to descriptive work. Everybody is invited to join the fun.

A second step that I think must be taken is more foundational. In section III it was explained that CDRT as it is described here must be a *representational* theory of language semantics, on pain of not getting the facts right. This was because it turned out that a linguistically acceptable text may have a representation K which has the same semantic value as a representation K' of an unacceptable text. Given that this is the case, we can account for the difference in acceptability only on the basis of the difference between K and K', not on the basis of a difference in semantic values. The same problem holds for DPL, DMG and some of my earlier work. The present solution of 'going representational' is mathematically impeccable as far as I can see, and some researchers, including Hans Kamp, have advocated the move for philosophical reasons. But for those who—like the present author—sympathise with Richard Montague's requirement that the level of intermediate representations should in principle be eliminable from the theory, things cannot remain as they stand now. The only road which is open to them is to adapt the theory, such that in all cases where K and K' represent texts which from a semantical point of view should be distinguished, K and K' really have a different semantic value. This means that it is either necessary to provide the underlying type logic with a more fine-grained notion of entailment (as it was done in Muskens [1995b] for different purposes), or that DRT must be grafted upon type logic in a slightly different way. However, such considerations do not affect the applicability of the present system and I am glad to leave them for future research.

APPENDIX: PROOFS OF PROPOSITIONS 2 AND 3

PROPOSITION 2. *If* $\mathbf{wp}(K, \top)$ *is closed then it is equivalent to* $\exists jK(i)(j)$ *for any i.*

PROOF. For any formula φ and state variable i let $(\varphi)^i$ denote the result of replacing each free individual variable x_k in φ with $\mathsf{V}(u_k)(i)$. We prove that

(a) $(\mathbf{tr}(\gamma))^i$ is equivalent to $\gamma(i)$
(b) $(\mathbf{wp}(K, \chi))^i$ is equivalent to $\exists j(K(i)(j) \wedge (\chi)^j)$

These statements are proved by induction on the complexity of DRT constructs. In the following $\approx$ stands for 'is equivalent to'.

(i) $(\mathbf{tr}(R\{\delta_1,\ldots,\delta_n\}))^i \approx (R(\delta_1{}^\dagger)\ldots(\delta_n{}^\dagger))^i \approx$ (by the definitions of $(.)^\dagger$ and $(.)^i$ and AX4) $R(\mathsf{V}(\delta_1)(i))\ldots(\mathsf{V}(\delta_n)(i)) \approx R\{\delta_1,\ldots,\delta_n\}(i)$
$(\mathbf{tr}(\delta_1 \textbf{ is } \delta_2))^i \approx (\delta_1{}^\dagger = \delta_2{}^\dagger)^i \approx \mathsf{V}(\delta_1)(i) = \mathsf{V}(\delta_2)(i) \approx \delta_1 \textbf{ is } \delta_2(i)$

(ii) $(\mathbf{tr}(\textbf{not } K))^i \approx (\neg\mathbf{wp}(K,\top))^i \approx \neg(\mathbf{wp}(K,\top))^i \approx \neg\exists j(K(i)(j) \approx (\textbf{not } K)(i)$

(iii) $(\mathbf{tr}(K_1 \textbf{ or } K_2))^i \approx (\mathbf{wp}(K_1,\top) \vee \mathbf{wp}(K_2,\top))^i \approx$
$(\mathbf{wp}(K_1,\top))^i \vee (\mathbf{wp}(K_2,\top))^i \approx \exists j(K_1(i)(j) \vee \exists j(K_2(i)(j) \approx (K_1 \textbf{ or } K_2)(i)$

(iv) $(\mathbf{tr}(K_1 \Rightarrow K_2))^i \approx (\neg\mathbf{wp}(K_1,\neg\mathbf{wp}(K_2,\top)))^i \approx \neg\exists j(K_1(i)(j) \wedge (\neg\mathbf{wp}(K_2,\top))^j) \approx$
$\neg\exists j(K_1(i)(j) \wedge \neg\exists k(K_2(j)(k)) \approx (K_1 \Rightarrow K_2)(i)$

(v) $(\mathbf{wp}([u_{k_1},\ldots,u_{k_n} \mid \gamma_1,\ldots,\gamma_m], \chi))^i$
$\approx (\exists x_{k_1}\ldots x_{k_n}(\mathbf{tr}(\gamma_1) \wedge\ldots\wedge \mathbf{tr}(\gamma_m) \wedge \chi))^i$
$\approx$ (by the Unselective Binding Lemma) $(\exists j(i[u_{k_1},\ldots,u_{k_n}]j \wedge$
$[\mathsf{V}(u_{k_1})(j)/x_{k_1},\ldots,\mathsf{V}(u_{k_n})(j)/x_{k_n}](\mathbf{tr}(\gamma_1) \wedge\ldots\wedge \mathbf{tr}(\gamma_m) \wedge \chi))^i$
$\approx$ (by the definition of $i[u_{k_1},\ldots,u_{k_n}]j$ and AX3)
$\exists j(i[u_{k_1},\ldots,u_{k_n}]j \wedge (\mathbf{tr}(\gamma_1) \wedge\ldots\wedge \mathbf{tr}(\gamma_m) \wedge \chi)^j)$
$\approx \exists j(i[u_{k_1},\ldots,u_{k_n}]j \wedge (\mathbf{tr}(\gamma_1))^j \wedge\ldots\wedge (\mathbf{tr}(\gamma_m))^j \wedge (\chi)^j)$
$\approx \exists j(i[u_{k_1},\ldots,u_{k_n}]j \wedge \gamma_1(j) \wedge\ldots\wedge \gamma_m(j) \wedge (\chi)^j)$
$\approx \exists j([u_{k_1},\ldots,u_{k_n} \mid \gamma_1,\ldots,\gamma_m](i)(j) \wedge (\chi)^j)$

(vi) $(\mathbf{wp}(K_1\,;K_2, \chi))^i \approx (\mathbf{wp}(K_1, \mathbf{wp}(K_2, \chi)))^i \approx$
$\exists k(K_1(i)(k) \wedge (\mathbf{wp}(K_2, \chi))^k) \approx \exists k(K_1(i)(k) \wedge \exists j(K_1(k)(j) \wedge (\chi)^j)) \approx$
$\exists j\exists k(K_1(i)(k) \wedge K_1(k)(j) \wedge (\chi)^j) \approx \exists j((K_1\,;K_2)(i)(j) \wedge (\chi)^j)$

The proposition directly follows from (b).

PROPOSITION 3. $K_1,\ldots,K_n \models_{DRT} K$ iff $\mathbf{wp}(K_1, \top),\ldots,\mathbf{wp}(K_n, \top) \models \mathbf{wp}(K, \top)$

PROOF. Suppose that $K_1,\ldots,K_n$ are true in i while K is not. Then, by (b) in the preceding proof, $(\mathbf{wp}(K_1,\top))^i,\ldots,(\mathbf{wp}(K_n,\top))^i$ hold while $(\mathbf{wp}(K,\top))^i$ does not. Use AX3 to choose an assignment a such that $a(x_k) = \mathsf{V}(u_k)(i)$ for all k; then $\mathbf{wp}(K_1,\top),\ldots,\mathbf{wp}(K_n,\top)$ are true under a while $\mathbf{wp}(K,\top)$ is false under a. Conversely, let $\mathbf{wp}(K_1,\top),\ldots,\mathbf{wp}(K_n,\top)$ be true under a and $\mathbf{wp}(K,\top)$ false under a. By AX1 and AX2 we can find an i such that $a(x_k) = \mathsf{V}(u_k)(i)$ for all free variables x_k in $\mathbf{wp}(K_1,\top),\ldots,\mathbf{wp}(K_n,\top),\mathbf{wp}(K,\top)$. Clearly, $(\mathbf{wp}(K_1,\top))^i,\ldots,(\mathbf{wp}(K_n,\top))^i$ hold but $(\mathbf{wp}(K,\top))^i$ is false and using (b) again we have that $K_1,\ldots,K_n$ are true in i while K is not.

REFERENCES

Asher, N., 1993, *Reference to Abstract Objects in Discourse,* Kluwer, Dordrecht.

Barwise, J.: 1987, Noun Phrases, Generalized Quantifiers and Anaphora, in P. Gärdenfors (ed.), *Generalized Quantifiers,* Reidel, Dordrecht, 1-29.

Van Benthem, J.F.A.K.: 1989, Semantic Parallels, in H.D. Ebbinghaus et al. (eds.), *Logic Colloquium. Granada 1987,* North-Holland, Amsterdam, 331-375.

Van Benthem, J.F.A.K.: 1991, *Language in Action,* North-Holland, Amsterdam.

Bittner, M.: 1994, Cross-Linguistic Semantics, *Linguistics and Philosophy* **17**, 53-108.

Bos, J., E. Mastenbroek, S. McGlashan, S. Millies and M. Pinkal, 1994, A Compositional DRS-based Formalism for NLP Applications, in H. Bunt, R. Muskens and G. Rentier (eds.), *Proceedings of the International Workshop on Computational Semantics,* ITK, Tilburg, 21-31.

Chierchia, G.: 1992, Anaphora and Dynamic Binding, *Linguistics and Philosophy* **15**, 111-183.

Chomsky, N.: 1981, *Lectures on Government and Binding,* Foris, Dordrecht.

Church, A.: 1940, A Formulation of the Simple Theory of Types, *The Journal of Symbolic Logic* **5**, 56-68.

Dekker, P.J.E.: 1993, *Transsentential Meditations,* Dissertation, University of Amsterdam.

Friedman, J., and D. Warren, 1980, Lambda Normal Forms in an Intensional Logic for English, *Studia Logica* **39**, 311-324.

Gabbay, D. and Günthner, F. (eds.): 1983, *Handbook of Philosophical Logic*, Reidel, Dordrecht.

Gallin, D.: 1975, *Intensional and Higher-Order Modal Logic*, North-Holland, Amsterdam.

Gazdar, G.: 1980, A Cross-Categorial Semantics for Coordination, *Linguistics and Philosophy* **3**, 407-409.

Goldblatt, R.: 1987, *Logics of Time and Computation,* CSLI Lecture Notes, Stanford.

Groenendijk, J. and Stokhof, M.: 1990, Dynamic Montague Grammar, in L. Kálmán and L. Pólos (eds.), *Papers from the Second Symposium on Logic and Language,* Akadémiai Kiadó, Budapest, 3-48.

Groenendijk, J. and Stokhof, M.: 1991, Dynamic Predicate Logic, *Linguistics and Philosophy* **14**, 39-100.

Heim, I., 1982, *The Semantics of Definite and Indefinite Noun Phrases,* Dissertation, UMass, Amherst, published in 1989 by Garland, New York.

Heim, I., 1983, 'File Change Semantics and the Familiarity Theory of Definiteness', in R. Bäuerle, C. Schwarze, and A. von Stechow, eds., *Meaning, Use and Interpretation of Language,* de Gruyter, Berlin.

Hendriks, H.: 1993, *Studied Flexibility: Categories and Types in Syntax and Semantics*, Doctoral dissertation, University of Amsterdam.

Henkin, L.: 1963, A Theory of Propositional Types, *Fundamenta Mathematicae* **52**, 323-344.

Janssen, T.: 1986, *Foundations and Applications of Montague Grammar,* CWI, Amsterdam.

Kamp, H.: 1981, A Theory of Truth and Semantic Representation, in J. Groenendijk, Th. Janssen, and M. Stokhof (eds.), *Formal Methods in the Study of Language, Part I,* Mathematisch Centrum, Amsterdam, 277-322.

Kamp, H. and Reyle, U.: 1993, *From Discourse to Logic,* Kluwer, Dordrecht.

Keenan, E. and Faltz, L.: 1978, *Logical Types for Natural Language*, UCLA Occasional Papers in Linguistics, 3.

Latecki, L. and M. Pinkal: 1990, *Syntactic and Semantic Conditions for Quantifier Scope,* Graduirung und Komparation, Arbeitspapier Nr. 13, Universität des Saarlandes.

Lewis, D.: 1975, Adverbs of Quantification, in E. Keenan (ed.), *Formal Semantics of Natural Language,* Cambridge University Press, 3-15.

May, R.: 1977, *The Grammar of Quantification,* Ph.D. dissertation, MIT, Cambridge.

Montague, R.: 1973, The Proper Treatment of Quantification in Ordinary English, in R. Montague, *Formal Philosophy*, Yale University Press, New Haven, 1974, 247-270.

Muskens, R.A.: 1989, Going Partial in Montague Grammar, in R. Bartsch, J.F.A.K. van Benthem and P. van Emde Boas (eds.), *Semantics and Contextual Expression,* Foris, Dordrecht, 175-220.

Muskens, R.A.: 1991, Anaphora and the Logic of Change, in Jan van Eijck (ed.), *JELIA '90, European Workshop on Logics in AI,* Springer Lecture Notes, Springer, Berlin, 414-430.

Muskens, R., 1994, Categorial Grammar and Discourse Representation Theory, *Proceedings of COLING 94,* Kyoto, 508-514.

Muskens, R.A.: 1995a, Tense and the Logic of Change, in: U. Egli, P.E. Pause, C. Schwarze, A. von Stechow and G. Wienold (eds.), *Lexical Knowledge in the Organisation of Language*, J. Benjamins, Amsterdam/Philadelphia, pp. 147-183.

Muskens, R.A.: 1995b, *Meaning and Partiality,* CSLI, Stanford CA.

Muskens, R.A., J.F.A.K. van Benthem, and A. Visser: forthcoming, Dynamics, in J.F.A.K. van Benthem and A. ter Meulen (eds.), *The Handbook of Logic in Linguistics,* Elsevier Science Publishers.

Partee, B. and Rooth, M.: 1983, Generalized Conjunction and Type Ambiguity, in R. Bäuerle, Ch. Schwarze and A. von Stechow (eds.), *Meaning, Use and Interpretation of Language,* De Gruyter, Berlin, 361-383.

Pratt, V.R.: Semantical Considerations on Floyd-Hoare Logic, *Proc. 17th IEEE Symp. on Foundations of Computer Science,* 109-121.

Rooth, M.: 1987, Noun Phrase Interpretation in Montague Grammar, File Change Semantics, and Situation Semantics, in P. Gärdenfors (ed.), *Generalized Quantifiers,* Reidel, Dordrecht, 237-268.

Saurer, W.: 1993, A Natural Deduction System for Discourse Representation Theory, *Journal of Philosophical Logic* **22**, 249-302.

Scott, D.: 1971, On Engendering an Illusion of Understanding, *Journal of Philosophy* **68**, 787-807.

Von Stechow, A.: 1974, ε–λ kontextfreie Sprachen: Ein Beitrag zu einer natürlichen formalen Semantik, *Linguistische Berichte* **34**, 1-33.

Zeevat, H.: 1989, A Compositional Approach to Discourse Representation Theory, *Linguistics and Philosophy* **12**, 95-131.

Reinhard Muskens
Department of Linguistics
Tilburg University
P.O. Box 90153, 5000 LE Tilburg
R.A.Muskens@kub.nl