CHAPTER SEVEN

SEMANTICS WITH LATTICES

7.1. BOOLEAN TYPES

In the type theory used in Montague's intensional logic (IL), all the types that get actually used as the target types for the translations of expressions of English, except type e, form Boolean algebras. Many of the computations in IL that we have learned to perform (semi-) automatically, like various λ-conversions, rely on (or alternatively, express) Boolean connections between different types. In the related theory of Boolean semantics of Keenan and Faltz (1985) all the types form Boolean algebras.

In this section I want to discuss some of the connections between Boolean types and discuss some of the differences between Montague's theory and the one presented in Keenan and Faltz.

Montague's extensional functional type theory was presented in Chapter One. Type e is interpreted as the domain of individuals, type t is the set of truth values $\{0, 1\}$, and every type $\langle a, b\rangle$ is the set of all functions from type a into type b. As Partee and Rooth 1983 observe, the only types that are actually used by Montague, besides type e, are types in the following set:

BOOL is the smallest set such that:

1. $t \in BOOL$
2. If $a \in TYPE$ and $b \in BOOL$ then $\langle a, b\rangle \in BOOL$

THEOREM. *Every type $b \in BOOL$ forms a complete atomic Boolean algebra.*

We have seen before that the set of truth values $\{0, 1\}$ with its standard truth functions defined on it forms a Boolean algebra, the minimal Boolean algebra. A fact that I mentioned in the last chapter is the following (I will state it here for Boolean algebras):

LIFTING

Let A be a non-empty set and let B be a complete atomic Boolean algebra, then the operations of B can be lifted on the function space, A^B (or $(A \rightarrow B)$ as I will write it here) by point-wise definition, and $(A \rightarrow B)$ forms a complete atomic Boolean algebra.

Point-wise definition means that you define the operations on the functions in terms of the operations on the values of those functions for every argument (i.e. in terms of the argument-value pairs, which we can think of as coordinates for points). So:

$f \wedge g$ is that function such that for every $a \in A$:
$f \wedge g(a) = f(a) \wedge_B g(a)$,

or, to use λ-abstraction in the metalanguage:

$f \wedge g := \lambda a.f(a) \wedge g(a)$
$f \vee g := \lambda a.f(a) \vee g(a)$
$\neg f := \lambda a.\neg f(a)$
$0 := \lambda a.0$ (the constant function on 0)
$1 := \lambda a.1$ (the constant function on 1)

It should be obvious that the function space $(A \rightarrow B)$ under these operations is indeed a Boolean algebra:
– Since it is the set of all functions from A into B, obviously, it contains all the functions defined above (so it is closed under the operations). (If, for some reason, we want types that do not contain all functions, but we do want to have point-wise definition, then of course we have to prove (or stipulate) that the set of functions that we do want is closed under lifting the operations in this way.)
– It is further trivial to check that the axioms for Boolean algebras hold for the function space. An example:

$$\begin{aligned}\neg(f \vee g) &= \neg(\lambda a.f(a) \vee g(a)) && \text{(by the above definition of } \vee\text{)}\\ &= \lambda a.\neg(f(a) \vee g(a)) && \text{(by the above definition of } \neg\text{)}\\ &= \lambda a.\neg f(a) \wedge \neg g(a) && \text{(by the fact that } B \text{ is a}\end{aligned}$$

Boolean algebra)

$= \lambda a.\neg f(a) \wedge \lambda a.\neg g(a)$ (by the above definition of $\wedge$)

$= \neg f \wedge \neg g$ (by, twice, the above definition of $\neg$)

–Furthermore, if we define, for set of functions F,

$$\wedge F = \lambda a.\wedge\{f(a): f \in F\}, \text{ and similarly } \vee F$$

then it is not hard to see that $(A \to B)$ is indeed complete.

What is left to prove is that $(A \to B)$ is atomic. Things are a bit more complicated here. Let AT_B be the set of atoms in B.

For $b \in AT_B$, $a \in A$, let us define:

$A_{a,b}$ is that function f such that:

1. $f(a) = b$
2. $f(x) = 0$ if $x \neq a$

So, $A_{a,b}$ is the function that maps a onto b and everything else onto 0.

We define $AT = \{A_{a,b}: a \in A \text{ and } b \in AT_B\}$ and we claim:

$(A \to B)$ is an atomic Boolean algebra with set of atoms AT.

Proof. We have to prove two things. First we have to prove that every $A_{a,b}$ is an atom, and secondly we have to prove that every function has an atom below it.

1. Suppose $f \leq A_{a,b}$. That means: $f \vee A_{a,b} = A_{a,b}$, i.e.

$$A_{a,b} = \lambda x.f(x) \vee A_{a,b}(x)$$

First we observe that for every argument x other than a, $f(x) = 0$. If not, then $f(x) \vee A_{a,b}(x) \neq 0$, and hence $f \vee A_{a,b} \neq A_{a,b}$.

So we have to look at the value for a. There are three possibilities.

(a) $f(a) = 0$. Then $f = \lambda a.0$.

(b) $f(a) = b$. Then $f = A_{a,b}$.

(c) $f(a) = x$, where x is neither 0 or b.

B is a Boolean algebra, b is an atom in B, so either $b \leq x$ or $b \leq \neg x$.

If $b \leq x$, then $f(a) \vee A_{a,b}(a) = x$, so $f \vee A_{a,b} \neq A_{a,b}$.

If $b \leq \neg x$, then $f(a) \vee A_{a,b}(a) = 1$, so $f \vee A_{a,b} \neq A_{a,b}$.

So we see that if $f \leq A_{a,b}$, then $f = \lambda a.0$ or $f = A_{a,b}$, so indeed $A_{a,b}$ is an atom in $(A \to B)$.

2. Every function (except 0) has an atomic function below it. This is again a matter of point-wise argumentation.

Take any function f that takes for some argument a non-zero value, pick any one of its function-value pairs, $\langle a, f(a)\rangle$ where $f(a)$ is non-zero.

Since B is atomic, $f(a)$ has an atom b below it. Replace $\langle a, f(a)\rangle$ by $\langle a, b\rangle$. Replace all the other values by 0. Call the resulting function g. Clearly $g \in AT$. Further $g \leq f$ (this should be obvious, if it isn't, check it). This concludes the proof.

In the particular case where B is $\{0, 1\}$, 1 is the atom. Hence, by the above definition, an atomic function in $A \to \{0, 1\}$ is a function mapping some element in A onto 1 and all the other elements onto 0, hence the atomic functions are the characteristic functions of the singleton sets of elements in A.

We have proved that $A \to \{0, 1\}$ is a complete atomic Boolean algebra. Its cardinality is $2^{|A|}$. Of course, that is the cardinality of pow$A$. Hence, up to isomorphism, $A \to \{0, 1\}$ and pow$A$ are identical. This is the algebraic background for the identification of sets with their characteristic functions.

Similarly, the cardinality of $(A \to (A \to \{0, 1\}))$ is $(2^n)^n = 2^{(n^2)}$, which is the cardinality of pow$(A \times A)$. So again, up to isomorphism $(A \to (A \to \{0, 1\}))$ and pow$(A \times A)$ are isomorphic.

Another connection that is important is the following. Let A be a complete atomic Boolean algebra. We know that A is isomorphic to some powerset Boolean algebra, hence for some n, the cardinality of A is 2^n. Look at $(A \to \{0, 1\})$. Obviously, its cardinality is 2^{2^n}. This means that $A \to \{0, 1\}$ is a *free complete* atomic Boolean algebra.

The above considerations tell us that every type in $BOOL$ is a complete atomic Boolean algebra. This means, that all these types correspond with powerset Boolean algebras.

Moreover, we see that if $a \in BOOL$ then $\langle a, t\rangle$ is a free complete atomic Boolean algebra. This tells us, in particular, that $\langle\langle e, t\rangle, t\rangle$, the type of noun phrases, is a free complete atomic Boolean algebra.

Let us look at the logical connectives $\neg$, $\wedge$, $\vee$. In Montague's theory, $\neg$, $\wedge$, $\vee$ are syncategorematic expressions that are introduced as the Boolean operations on type t, let us write this as $\neg_{[t]}$, $\wedge_{[t]}$, $\vee_{[t]}$. Clearly,

we can introduce them as specially chosen functions in the correct types as well:

$$\neg_{[t]} \in \langle t, t\rangle; \quad \wedge_{[t]}, \vee_{[t]} \in \langle t, \langle t, t\rangle\rangle.$$

Now, of course, the connectives are not just restricted to type t, but can conjoin expressions of the same type in many more types, typically in all Boolean types.

Given that all Boolean types are Boolean algebras, we can thus introduce the Boolean operations in a schema:

$\forall a \in BOOL$: $\neg_{[a]}, \wedge_{[a]}, \vee_{[a]}$ are the Boolean operations on a (hence, $\neg_{[a]} \in \langle a, a\rangle$).

Montague, in Montague (1973), at first sight follows a different approach (he does so to some extent implicitly; Partee and Rooth and others make this approach explicit). He takes the Boolean operations on type t as basic and defines the operations on other types in terms of that.

Given the lifting theorem, stated above, there is no incompatibility between these two approaches. Every type has, of course, exactly one set of Boolean connectives on it. Whether we just stipulate that $\neg_{[a]}$ is Boolean negation at type a, or whether we define $\neg_{[a]}$ in terms of Boolean negation at lower types, if we can do the latter, we get the same negation $\neg_{[a]}$.

The lifting theorem tells us that indeed for all Boolean types we can do this. Hence we can give a definition schema for the Boolean operations at all conjoinable types:

Generalized Negation, Conjunction, Disjunction

The definition is inductive:

1. $\neg_{[t]}, \wedge_{[t]}, \vee_{[t]}$ are the standard truth functions.
2. If a is a Boolean type of the form $\langle b, c\rangle$ then $\neg_{[a]}$, $\wedge_{[a]}$, $\vee_{[a]}$ are lifted from c onto a $(=\langle b, c\rangle)$ by pointwise definition.

Formally, where $a = \langle b, c\rangle$ and $\mathbf{P}, \mathbf{Q} \in a$, $P \in b$:

$$\neg_{[a]} := \lambda\mathbf{P}\lambda P.\neg_{[c]}\mathbf{P}(P)$$

Here $\mathbf{P}(P)$ is of type c, so is $\neg_{[c]}\mathbf{P}(P)$, so $\lambda P.\neg_{[c]}\mathbf{P}(P)$ is of type a, so $\neg_{[a]}$ is indeed of type $\langle a, a\rangle$.

$$\wedge_{[a]} := \lambda\mathbf{Q}\lambda\mathbf{P}\lambda P.(\wedge_{[c]}(\mathbf{Q}(P)))(\mathbf{P}(P))$$

or more perspicuously:

$$\wedge_{[a]} := \lambda\mathbf{Q}\lambda\mathbf{P}\lambda P.\mathbf{P}(P) \wedge_{[c]} \mathbf{Q}(P)$$
$$\vee_{[a]} := \lambda\mathbf{Q}\lambda\mathbf{P}\lambda P.\mathbf{P}(P) \vee_{[c]} \mathbf{Q}(P)$$

In this section I will use the following variables:

x, y are variables of type e
P, Q are variables of type $\langle e, t\rangle$
R, S are variables of type $\langle e, \langle e, t\rangle\rangle$
T is a variable of type $\langle\langle e, t\rangle, t\rangle$

Furthermore, in the following example $\mathbf{P}$ is a variable of type $\langle\langle e, t\rangle, \langle e, t\rangle\rangle$.

Example:

$$\neg_{[\langle e,t\rangle]} = \lambda P\lambda x.\neg_{[t]} P(x)$$

So $\neg(Walk) = \lambda x.\neg Walk(x)$

$$\neg_{[\langle\langle e,t\rangle,t\rangle]} = \lambda T\lambda P.\neg_{[t]} T(P)$$
$$\neg_{[\langle\langle e,t\rangle,\langle e,t\rangle\rangle]} = \lambda\mathbf{P}\lambda P.\neg_{[\langle e,t\rangle]}\mathbf{P}(P)$$

Hence

$$\neg_{[\langle\langle e,t\rangle,\langle e,t\rangle\rangle]} = \lambda\mathbf{P}\lambda P.(\lambda Q\lambda x.\neg_{[t]} Q(x))\,(\mathbf{P}(P))$$

converting $\mathbf{P}(P)$ for Q gives:

$$= \lambda\mathbf{P}\,\lambda P\lambda x.\neg_{[t]}(\mathbf{P}(P))(x)$$

So

$$\neg(Slowly) = \lambda P\lambda x.\neg(Slowly(P))(x)$$

Hence,

$$\neg(Slowly)(Walk) = \lambda x.\neg(Slowly(Walk))(x)$$

Let us now look at the connection between e and $\langle\langle e, t\rangle, t\rangle$.

Montague associates with a proper name *John* a constant j of type e. However, he does not translate *John* as j, but as $\lambda P.P(j)$ of type $\langle\langle e, t\rangle, t\rangle$, the set of all properties of j. The reason is well known: terms like *every man* cannot be of type e, but have type $\langle\langle e, t\rangle, t\rangle$. We cannot

conjoin an individual with a set of properties, so for conjunction *John and every man*, we have to assume that *John* has type $\langle\langle e, t\rangle, t\rangle$ as well. Conjunction at that type gives us:

$$\lambda P.P(j) \wedge_{[\langle\langle e,t\rangle, t\rangle]} \lambda P.\forall x[Man(x) \rightarrow P(x)]$$

which is equivalent to:

$$\lambda P.P(j) \wedge \forall x[Man(x) \rightarrow P(x)].$$

We have argued in the previous chapter that we don't lose any information by lifting j in this way from type e to type $\langle\langle e, t\rangle, t\rangle$.

$\langle\langle e, t\rangle, t\rangle$ is isomorphic to powpow$D$ (Where D is the set of individuals). The isomorphism is given through the bijection that maps every individual $x \in D$ onto the corresponding ultrafilter x^* in powpow$D$, this is $\lambda P.P(x)$. If D itself had been a Boolean algebra,* would have been an embedding.

We see, thus, two properties of the principle connecting individuals with their generated ultrafilters (or, in the type shifting perspective of Partee and Rooth (1983): of the rule shifting expressions of type e to expressions of type $\langle\langle e, t\rangle, t\rangle$):

1. it preserves the structure of the lower domain (if any) into the higher domain.
2. it preserves meaning by the following correspondence:

$$\forall P\forall x[P(x) \leftrightarrow x^*(P)].$$

Such preservation of structure and meaning is a salient feature of other cases as well, where, for reasons of conjoinability, an expression is given a translation in a higher type than needed if we didn't look at conjoinability.

In fact, this preservation is the guiding principle that leads you to the definition of the translation of the expression in the higher type. You shift a certain meaning from a type where you cannot perform a certain operation to a type where you can. However, in an important intuitive sense you would want to say that the shifted meaning is the same as the original, but adjusted for the new type: what it does at the higher type when it combines with another meaning corresponds to what it would do at the lower type when it combines there with whatever corresponds to this other meaning *if there is anything that corresponds to this other meaning*.

In the example above, we do not arbitrarily decide to map x onto x^*, we do that *because* * is the unique operation that satisfies: $\forall P\forall x[P(x) \leftrightarrow x^*(P)]$.

As I said above, Keenan and Faltz (1985) assume only Boolean types. They do not have a type e. Their lowest type is type $\langle e, t\rangle$. Well, in fact it isn't: they call it type p, for properties, and they take properties as primitives, but since they assume that p is a complete atomic Boolean algebra of properties, we know that p is isomorphic to $(D \rightarrow \{0, 1\})$ for some set D, hence we could as well call it $\langle e, t\rangle$.

Keenan and Faltz can work without a domain of individuals, because by the above considerations, they don't have to assume such a domain and still get individuals. We have seen that for all practical purposes individuals x and their corrsponding ultrafilters x^* behave in the same way. In Keenan and Faltz's theory the domain of noun phrases also is $\langle\langle e, t\rangle, t\rangle$ (that is, $\langle p, t\rangle$). We know that this is a free complete atomic Boolean algebra generated by the ultrafilters of properties (i.e. the x^*s). Keenan and Faltz take those ultrafilters to be the individuals to start with.

The only difference, then, at this level, is a notational one. In Montague's theory you can define the interpretation of *John* as $j^* := \lambda P.P(j)$. In Keenan and Faltz', theory you know that j^* is the set of all properties of some 'real' individual, but you can't express that, because you don't have that 'real' individual in your model.

We will now look at one-place predicates. In Montague's (extensional) theory, intransitive verbs are interpreted as one-place predicates of type $\langle e, t\rangle$, so they apply to individuals. Since Montague lifts the interpretations of proper names from type e to type $\langle\langle e, t\rangle, t\rangle$ and interprets all noun phrases in this type, intransitive verbs can no longer apply to the subject noun phrase. Consequently, Montague switches the function-argument structure around and assumes that in a sentence the subject applies to the intransitive verb. This leads to a well known asymmetry between subjects and objects of transitive verbs: transitive verbs apply to their objects, but not to their subjects.

Keenan and Faltz (and others) object to this aspect of Montague's theory and assume a uniform function-argument hypothesis: verbs are functions on all their syntactic arguments. From this it follows that they have to interpret intransitive verbs as being of type $\langle\langle\langle e, t\rangle, t\rangle, t\rangle$, or $\langle T, t\rangle$ for short.

We are interested, then, in lifting predicates of type $\langle e, t\rangle$ to predicates

of type $\langle T, t\rangle$. Let us look at the relations between $\langle e, t\rangle$ and $\langle T, t\rangle$. We know that T is a free Boolean algebra. Let U be the set of all ultrafilters in T and let $\langle U, t\rangle$ be the set of all functions from U into t. Since * is a bijection between e and U, obviously * is an isomorphism between $\langle e, t\rangle$ and $\langle U, t\rangle$.

In fact, with some notational fantasy, $\langle U, t\rangle$ is defined as:

$$\langle U, t\rangle := \{\lambda x^*.P(x)\colon P \in \langle e, t\rangle\}.$$

Clearly, for every y, P: $\lambda x^*.P(x)\ (y^*) \leftrightarrow P(y)$.

The operations on $\langle U, t\rangle$ are lifted from t:

$$\neg(\lambda x^*.P(x)) := \lambda x^*.\neg P(x)$$
$$(\lambda x^*.P(x)) \wedge (\lambda x^*.Q(x)) := \lambda x^*.P(x) \wedge Q(x)$$
$$(\lambda x^*.P(x)) \vee (\lambda x^*.Q(x) := \lambda x^*.P(x) \vee Q(x)$$

We are interested in finding functions in $\langle T, t\rangle$ that correspond in an intuitive way with the functions in $\langle e, t\rangle$. This gives us two conditions:

1. Every such function in $\langle T, t\rangle$ should be in one–one correspondence with a property P in $\langle e, t\rangle$, that is, this domain should be of the form $\{P^*\colon P \in \langle e, t\rangle\}$ where * is a bijection between $\langle e, t\rangle$ and this domain of functions.
2. P^* should be an *extension* of $\lambda x^*.P(x)$, because only that will guarantee that: for all y: $P^*(y^*) \leftrightarrow P(y)$.

Now:

$$\langle U, t\rangle = \{\lambda x^*.P(x)\colon P \in \langle e, t\rangle\} = \{\lambda x^*.x^*(P)\colon P \in \langle e, t\rangle\}$$

What we need to find is some function of P that extends $\lambda x^*.x^*(P)$. The obvious candidate is: $\lambda T.T(P)$.

Let * be the function mapping P onto $\lambda T.T(P)$. Clearly * is an injection: if $P \neq Q$ then $P^* \neq Q^*$. Moreover, for all x: $P^*(x^*) \leftrightarrow \lambda T.T(P)\ (x^*) \leftrightarrow x^*(P) \leftrightarrow P(x)$. So let us define:

$$\mathbf{P}^* = \{\lambda T.T(P)\colon P \in \langle e, t\rangle\}$$

That $\mathbf{P}^*$ is a domain that does what we want it to do can also be seen in another way.

We said that we were interested in a domain of functions in $\langle T, t\rangle$ that extend the functions in $\langle U, t\rangle$. An obvious domain of functions that does exactly that is given by the definition of free algebras: by definition,

given a free algebra F and an algebra B: every function from the generators of F into B can be uniquely extended to a homomorphism from F into B. U is the set of generators in T, hence every function f in $\langle U, t\rangle$ is in one–one correspondence with a homomorphism in $\langle T, t\rangle$ extending f. This means that there are as many homomorphisms in $\langle T, t\rangle$ as there are functions in $\langle U, t\rangle$, and hence as there are functions in $\langle e, t\rangle$. Thus the class of homomorphisms in $\langle T, t\rangle$, $\hom(\langle T, t\rangle)$ is a class that is naturally suited for our purposes.

Now note that $\hom(\langle T, t\rangle)$ and our set $\mathbf{P}^*$ have the same cardinality and both consist of functions that uniquely extend functions in $\langle U, t\rangle$. This is not surprising, since we can easily show that: (where J is of type T)

$$\begin{aligned}\lambda T.T(P)(\neg_{[T]}J) &= (\neg_{[T]}J)(P)\\ &= ((\lambda T.\lambda Q.\neg_{[t]}T(Q))(J))\,(P)\\ &= \lambda Q.\neg_{[t]}J(Q)\,(P) = \neg_{[t]}J(P)\\ &= \neg_{[t]}(\lambda T.T(P)(J))\end{aligned}$$

The same for the other connectives.

This means that every function of the form $\lambda T.T(P)$ is in fact a homomorphism in $\langle T, t\rangle$.

Every function $\lambda T.T(P)$ extends $\lambda x^*.x^*(P)$. Since for every $\lambda x^*.x^*(P)$ there is a unique homomorphism extending it, we know that this unique homomorphism is $\lambda T.T(P)$.

Now obviously every homomorphism $h \in \hom(\langle T, t\rangle)$ extends some function $\lambda x^*.x^*(P)$ (just take the restriction of H to $\langle U, t\rangle$). What follows is:

1. $\mathbf{P}^* = \{\lambda T.T(P)\colon P \in \langle e, t\rangle\} = \hom(\langle T, t\rangle)$.
2. $\mathbf{P}^*$ has the same cardinality as $\langle e, t\rangle$
3. $\forall x \forall P\colon P^*(x^*) \leftrightarrow P(x)$

Consequently, we can translate the intransitive verb *Walk* as *Walk** and *John* as j^*, and assume that *John Walks* translates as:

$$\begin{aligned}Walk^*(j^*) &= \lambda T.T(Walk)\ (\lambda P.P(J))\\ &= (\lambda P.P(J))\,(Walk) = Walk(j)\end{aligned}$$

Similarly, *Every man walks* will be translated as:

$$\lambda T.T(Walk)\ (\lambda P.\forall x[Man(x) \rightarrow P(x)]),$$

which is equivalent to:

$$\forall x[Man(x) \rightarrow Walk(x)].$$

P*, thus, is excellently suited to form the target for translating intransitive verbs. Note that the principle lifting predicates from $\langle e, t\rangle$ to $\langle T, t\rangle$ has exactly the same form as the principle lifting terms from e to T.

We have now lifted the verb phrases from $\langle e, t\rangle$ into $\langle T, t\rangle$. Let us now consider the connectives at this higher type. What should $\neg P^*$ be?

Earlier, when we considered lifting from e to T the answer was clear: $\{x^*: x \in e\}$, the range of e under lifting, has no structure to it. More importantly, there are elements of T that are not in e^*, that are the interpretations of natural language expressions, like quantified expressions, and that can be conjoined with the lifted elements of e.

This leaves us no choice but to take for the domain of noun phrases the whole type T, and hence the Boolean operations are just the Boolean operations on T, that is, the operations lifted from t onto $\langle\langle e, t\rangle, t\rangle$.

In the case of $\langle T, t\rangle$, the situation is somewhat different. If we could argue that there are intransitive verbs that cannot be regarded as lifted from some predicate of type $\langle e, t\rangle$, i.e. verbs that do not correspond to any predicate of type $\langle e, t\rangle$, our case would be made, we would have to assume that the domain of intransitive verbs is the whole domain $\langle T, t\rangle$. Such a verb would have to be a verb V that is not in **P***. This means that V is not a homomorphism on its argument.

Such verbs are very hard to find. Only two kinds of examples have been provided in the literature, and both are disputed.

Ballmer 1982 presents the case of *collective verbs*. *John and Bill meet* is not equivalent to *John meets and Bill meets*. If we analyze this as *Meet*($j^* \wedge b^*$), then *Meet* is not a homomorphism.

However, as we will see in the next section, we don't have to assume that *John and Bill* in that example is represented as $j^* \wedge b^*$. If we give structure to the domain of 'real' individuals, and assume that we have plural individuals like $j + b$, the sum of j and b, then we represent the example as: *Meet**$((j + b)^*)$ and the argument collapses.

The other argument is a theoretical argument. Following Partee and Rooth (1983), Hendriks (1987) develops a theory where type shifting principles like the ones we are discussing here are used as a mechanism to deal with scope ambiguities.

To get the two readings of *Every man doesn't walk*, Hendriks assumes that we have two strategies:

1. Start with *Walk* of type $\langle e, t\rangle$; apply negation at type $\langle e, t\rangle$, which gives us $\lambda x.\neg Walk(x)$; lift this to type $\langle T, t\rangle$: $\lambda T.T(\lambda x.\neg Walk(x))$ and apply it to $\lambda P.\forall x[Man(x) \rightarrow P(x)]$, which gives us the reading with *every* wide scope over negation.
2. Start with *Walk* of type $\langle e, t\rangle$; lift it to type $\langle T, t\rangle$, which gives us $\lambda T.T(Walk)$; *apply negation at type* $\langle T, t\rangle$, which gives us $\lambda T.\neg T(Walk)$; and apply that to $\lambda P.\forall x[Man(x) \rightarrow P(x)]$, which gives us the reading with *every* narrow scope under negation.

This theory, thus, makes essential use of the predicate $\lambda T.\neg T(Walk)$, which is not in **P***.

Hendriks' theory is elegant and extends naturally to scope ambiguities between different arguments of a verb, but it is not clear that it is the correct way of analyzing scope ambiguities.

In the first place it overgenerates in ways that Montague's analysis of scope ambiguities doesn't. It has to assume that *Every woman walks or talks* has a reading that is equivalent to *Every woman walks or every woman talks*. This is because we can lift both *Walk* and *Talk* to $\langle T, t\rangle$, apply disjunction there and get $\lambda T.T(Walk) \vee T(Talk)$. This we can apply to the translation of *every man* and we get the reading where *every man* distributes over the disjunction. Such a reading is strongly disputed, however.

Secondly, it undergenerates in ways that Montague's analysis doesn't. On the type shifting analysis *a woman* gets wide scope in *Every man loves a woman* by turning *loves a woman* into a predicate that gives the argument that it takes narrow scope under *a woman* $(\lambda T.[\lambda P.\exists x[Woman(x) \wedge P(X)]\ (\lambda y.T(\lambda x.love(x, y))])$. Now take the following example:

Every man loves a woman or hates a dog.

On Montague's analysis, it is very easy to give both *a woman* and *a dog* wide scope over *every man*, and get the possible reading:

> there is a woman and there is a dog such that every man loves her or hates it.

The type shifting analysis can only give *a woman* and *a dog* wide

scope over *every man*, by first both shifting *loves a woman* and *hates a dog* in the above way, then apply disjunction at the level $\langle T, t\rangle$. But then, as we have seen, necessarily *every man* will distribute over the disjunction and the reading will be:

> there is a woman such that every man loves her or there is a dog such that every man hates it.

These are serious problems and until they are properly dealt with, we cannot accept the type shifting analysis of scope ambiguities as an argument for recognizing verbs (or verb meanings in this case) that are not homomorphisms.

Keenan and Faltz assume a more standard analysis of scope ambiguities and assume that there are no verbs that have non-homomorphisms as their interpretation.

If we assume the full domain $\langle T, t\rangle$ as the interpretation domain for intransitive verbs, this domain contains many elements that will never be the interpretation of any natural language expression.

The ideology of Keenan and Falt is to restrict the domains of interpretation to contain only those elements that can possibly be the interpretations of natural language expressions. Hence they assume that the type of intransitive verbs is not the full domain $\langle T, t\rangle$, but only $\mathbf{P}^*$.

This means that the lifted Boolean operations on $\langle T, t\rangle$ are not available for them. Yet, obviously, the domain of one place predicates has to have a Boolean structure. This means that we have to impose a Boolean structure upon $\mathbf{P}^*$ itself.

We cannot use the standard lifted connectives here because, as I said earlier, $\mathbf{P}^*$ is not closed under the standard lifted connectives. This can be seen easily.

Let $P \in \langle e, t\rangle$. Consider $\neg_{[\langle T,t\rangle]}\ (\lambda T.T(P))$. This is:

$$\lambda T.\neg_{[t]} T(P).$$

Question: is $\lambda T.\neg T(P) \in \mathbf{P}^*$?

By definition of $\mathbf{P}^*$ it would be only if there is some $Q \in \langle e, t\rangle$ such that $\lambda T.\neg_{[t]} T(P) = \lambda T.T(Q)$ and this holds iff for all generalized quantifiers $\mathbf{Q}$: $\neg\mathbf{Q}(P) \leftrightarrow \mathbf{Q}(Q)$.

One of these generalized quantifiers is the domain $\langle e, t\rangle$ itself (which corresponds to pow$D$, the set of all subsets of D), so the above statement should hold for $\langle e, t\rangle$, that is, in set terms, there has to be some

Q, such that $P \notin \text{pow}D$ iff $Q \in \text{pow}D$. Since both P and Q are of type $\langle e, t\rangle$ this can never be true, hence $\lambda T.\neg T(P) \notin \mathbf{P}^*$.

So the question is: can we impose a Boolean structure upon $\mathbf{P}^*$ itself? The answer is obviously yes: $\mathbf{P}^*$ has the same cardinality as $\langle e, t\rangle$. $\langle e, t\rangle$ is a Boolean algebra, so why couldn't we make $\mathbf{P}^*$ one. Even more, since there is exactly one complete atomic Boolean algebra of the cardinality of $\langle e, t\rangle$, we automatically make $\langle e, t\rangle$ and $\mathbf{P}^*$ isomorphic, if we turn $\mathbf{P}^*$ into a complete atomic Boolean algebra.

However, we are constrained by yet another natural preservation requirement. The functions in $\mathbf{P}^*$ correspond one–one with the functions in $\langle e, t\rangle$. The isomorphism h between $\langle e, t\rangle$ and $\mathbf{P}^*$ has to map every $P \in \langle e, t\rangle$ onto $\lambda T.T(P)$. If the Boolean operations on $\mathbf{P}^*$ are *not*, *and*, and *or*, then h: $\langle e, t\rangle \rightarrow \mathbf{P}^*$ has to be a bijection such that:

1. $\forall P \in \langle e, t\rangle$: $h(P) = \lambda T.T(P)$
2. $h(\neg_{[\langle e,t\rangle]} P) = not\ h(P)$
   $h(P \wedge Q) = h(P)\ and\ h(Q)$
   $h(P \vee Q) = h(P)\ or\ h(Q)$

Given this, we read the definitions of the connectives *not*, *and*, *or* on $\mathbf{P}^*$ directly off the above equations:

$$not(\lambda T.T(P)) = \lambda T.T(\neg P)$$
$$(\lambda T.T(P))\ and\ (\lambda T.T(Q)) = \lambda T.T(P \wedge Q)$$
$$(\lambda T.T(P))\ or\ (\lambda T.T(Q)) = \lambda T.T(P \vee Q)$$

The functions on the right hand side of these equations are clearly in $\mathbf{P}^*$ $(=\{\lambda T.T(P): P \in \langle e, t\rangle\})$. So indeed $\mathbf{P}^*$ is closed under these operations.

Clearly, then, * is an isomorphism between $\langle\langle e, t\rangle, \neg, \wedge, \vee\rangle$ and $\langle \mathbf{P}^*, not, and, or\rangle$.

Note, by the way, that $\langle \mathbf{P}^*, not, and, or\rangle$ is not a subalgebra of $\langle\langle T, t\rangle, \neg, \wedge, \vee\rangle$. It is a Boolean algebra on a subset of $\langle T, t\rangle$.

Let us look at relations. If a relation like *love* has to be interpreted as a function on all its syntactic arguments, we have to lift it from type $\langle e, \langle e, t\rangle\rangle$ to type $\langle T, \langle T, t\rangle\rangle$. Now we can make exactly the same argument as before. Let's look at $\langle U, \langle U, t\rangle\rangle$.

$$\langle U, \langle U, t\rangle\rangle := \{\lambda y^* \lambda x^*.R(x, y): R \in \langle e, \langle e, t\rangle\}.$$

Now:

$$\langle U, \langle U, t\rangle\rangle := \{\lambda y^* \lambda x^*.R(x, y): R \in \langle e, \langle e, t\rangle\}$$
$$= \{\lambda y^* \lambda x^*.x^*(\lambda x.y^*(\lambda y.R(x, y))): R \in \langle e, \langle e, t\rangle\}$$

Since T is a free Boolean algebra and $\langle T, t\rangle$ is a complete atomic Boolean algebra, we know that every such function can be extended to a unique homomorphlsm in $\langle T, \langle T, t\rangle\rangle$.

It should come as no surprise that the set of these homomorphisms is exactly:

$$\mathbf{R}^* = \{\lambda T.\lambda T'.T'(\lambda x.T(\lambda y.R(x, y))): R \in \langle e, \langle e, t\rangle\rangle\}$$

If we translate *love* as

$$love^* = \lambda T.\lambda T'.T'(\lambda x.T(\lambda y.love(x, y)))$$

then *Every man loves a woman* will translate as:

$$(love^*(a\ woman))(every\ man)$$

which reduces to:

$$\forall x[Man(x) \rightarrow \exists y[Woman(y) \wedge Love(x, y)]]$$

Now there is a further observation to be made:

$$\text{for any } Q \in T,\ R^* \in \mathbf{R}^*: R^*(Q) \in \mathbf{P}^*.$$

This holds because $\lambda x.Q(\lambda y.R(x, y)) \in \langle e, t\rangle$. So the relations are homomorphisms from T into $\langle T, t\rangle$ and yield, for any argument a homomorphism in $\langle T, t\rangle$, that is, an element of $\mathbf{P}^*$.

From here, the positions are the same. If we want to define the connectives, we have a choice of either taking the full domain $\langle T, \langle T, t\rangle\rangle$ and the connectives lifted ultimately from t or we can define a Boolean structure on $\mathbf{R}^*$ itself, if we don't want to allow relations that are not interpreted as homomorphisms. The latter is what Keenan and Faltz do.

There are now two equivalent ways of defining these operations. In the first place, we can define them directly:

$$not(\lambda T.\lambda T'(\lambda x.T(\lambda y.T'.R(x, y))))$$
$$= \lambda T.\lambda T'.T'(\lambda x.T(\lambda y.\neg R(x, y)))$$

$$\lambda T.\lambda T'.T'(\lambda x.T(\lambda y.R(x, y))) \quad and$$
$$\lambda T.\lambda T'.T'(\lambda x.T(\lambda y.S(x, y)))$$
$$= \lambda T.\lambda T'.T'(\lambda x.T(\lambda y.R(x, y) \wedge S(x, y)))$$
$$\lambda T.\lambda T'.T'(\lambda x.T(\lambda y.R(x, y))) \quad or$$
$$\lambda T.\lambda T'.T'(\lambda x.T(\lambda y.S(x, y)))$$
$$= \lambda T.\lambda T'.T'(\lambda x.T(\lambda y.R(x, y) \vee S(x, y)))$$

Secondly, since $\mathbf{P}^*$ is a Boolean algebra under the internal operations and $\mathbf{R}^*$ is a set of functions from T into $\mathbf{P}^*$, we can lift the operations from $\mathbf{P}^*$ onto $\mathbf{R}^*$ (if $\mathbf{R}^*$ is closed under them, which it is):

$$not_R(\lambda T.\lambda T'.T'(\lambda x.T(\lambda y.R(x, y)))$$
$$= \lambda T.not_P(\lambda T'.T'(\lambda x.T(\lambda x.R(x, y)))$$
$$= \lambda T.\lambda T'.T'(\lambda x.T(\neg_{[\langle e,t\rangle]} \lambda y.R(x, y)))$$
$$= \lambda T.\lambda T'(\lambda x.T(\lambda y.T'(\lambda y.\neg R(x, y)))$$

Similarly for *and* and *or*.

This forms the basis for Keenan and Faltz' type theory (at least, the types for relations between individuals):

The type of one-place predicates is $\mathbf{P}^*$ (the set of homomorphisms from T into $\{0, 1\}$).

The type of two-place predicates is $\mathbf{R}^*$ (the set of homomorphisms from T into $\mathbf{P}^*$)

The type of three-place predicates is the set of homomorphisms from T into $\mathbf{R}^*$, etc.

Note that once we have restricted ourselves to Keenan and Faltz types, the above procedure for defining the connectives will work at any relational type. Hence generalized conjunction, disjunction and negation are defined in the Keenan and Faltz theory by first lifting the operations of p $(=\langle e, t\rangle)$ internally from the type p of properties of individuals to the set of one place relations (i.e. *not* $\lambda T.T(P) = \lambda T.T(\neg P)$) and from there on pointwise lifting them onto the other types.

7.2. PLURALS

I already briefly mentioned the problem of collective predicates in the previous section. In a sentence like *John and Bill and Henry and George meet*, the predicate *meet* does not distribute over the conjuncts, unlike *are boys* in *John and Bill and Henry and George are boys*. This is shown by the fact that the second sentence is equivalent to *John and Bill are boys and Henry and George are boys*, but such an equivalence does not hold for *meet*. *Meet* is a collective predicate, *are boys* a distributive predicate. Other predicates, like *carry a piano upstairs* are taken to be ambiguous between a collective and a distributive reading.

The problem of collective predicates is not just related to conjunction, the plural definite shows exactly the same behaviour: e.g. *The kids meet* does not mean that every kid meets, while *The kids are boys* does mean that every kid is a boy. In fact, the similarity is so great, that we will want to treat these cases, conjunction and plural definites, on a par.

The task then, is to provide a semantics for plurality that can deal with these phenomena.

Such theories have been around since Bennett (1975). One major innovation in the theory in Link (1983) (which in many respects is similar to the theory in Scha, 1981) is that unlike most previous theories, singular and plural terms are treated on a par. There is no ontological difference between the denotations of singular terms like *the boy* and plural terms like *the boys*. They are all part of one domain of singular and plural individuals.

This domain comes equiped with a part-of relation, or alternatively a sum operation, and the difference between singular individuals and plural individuals is that the latter, but not the first, have other (in particular singular) individuals as parts, or alternatively: plural individuals are sums of singular individuals: *John and Bill* denotes the plural individual that is the sum of the singular individuals John and Bill; *the Boys* denotes the plural individual that is the sum of the singular boys.

There will be predicates that just take singular individuals (singular predicates like *is a boy*) and predicates that take just plural individuals (collective predicates like *meet*) in their extension.

Note that already this much gives us predicates that do not distribute: if the denotation of *John and Bill*, the sum of John and Bill, is an individual in its own right that can be in the extension of a predicate like *meet*, there is no reason to expect that it would follow from this

that the individuals making up that sum would be in the extension of that predicate as well. The question then becomes: how can we deal with distributive predicates?

I will here present a version of Link's theory that is very close to the one in Link (1983). I will not here motivate the basic aspects of the theory more than I did above. For an introduction, a critical discussion and an alternative, see Landman (1989).

I will first discuss some general points about the structures and then give the semantics for a language with plurals. I will completely ignore here the problems of so-called group interpretations, for that, see Link (1984), [forthcoming] and Landman (1989). Further, I will postpone the discussion of mass nouns to the next section.

We are interested, thus, in domains of singular and plural individuals, where plural individuals are sums of singular individuals, and singular individuals are not themselves sums (except of themselves). We have introduced structures that are ideally suited for this: *i*-join semilattices: structures $\langle A, \bigvee \rangle$ with a complete* join operation. The discussion in the previous chapter was already a prelude to the present discussion. I argued there that we are not just interested in any structure of this form, but only in free structures: *i*-join semilattices, freely generated under sum by a set of minimal elements.

I argued in the previous chapter that in a sense, such structures are exactly the complete atomic Boolean algebras with the 0 element cut out.

The sense in which they are the same is that they are the same structures *geometrically*: they have the same diagrams in that the very same diagram can be interpreted either as a free *i*-join semilattive with sum and (by adding a 0) as a complete atomic Boolean algebra with all the Boolean operators.

Geometrically the structures are the same, algebraically they are not. Remember the distinction between Boolean lattices and Boolean algebras: the situation is the same here.

The difference shows up, for instance, when we look at homomorphisms. A homomorphism from $\langle A, \bigvee \rangle$ into $\langle A', \bigvee \rangle$ is a function respecting $\bigvee$. A homomorphism from $\langle B, \neg, \wedge, \vee, \bigwedge, \bigvee, 0, 1 \rangle$ into $\langle B', \neg, \wedge, \vee, \bigwedge, \bigvee, 0, 1 \rangle$ is a function respecting $\neg, \wedge, \vee, \bigwedge, \bigvee, 0, 1$.

Any Boolean homomorphism is of course a join semilattice homomorphism, but not vice versa: the latter are Boolean *join homomorphisms*.

Link endowes his domains with the structure of a complete atomic Boolean algebra, rather than a free *i*-join semilattice. Though it may seem pedantic to distinguish between structures that are so much the same, I will briefly give a few reasons to prefer working with free *i*-join semilattices.

In the first place, it makes some clauses more elegant. In a full Boolean algebra we have this 0 element. In all the important definitions though, we want to apply our concepts to singular or plural individuals, excluding 0. This means that we have to add exclusion clauses ($x \neq 0$) to all of them. Assuming from the start that 0 is not there, will make the definitions simpler and more readable.

The second reason is not just esthetic, though. Our operation $\vee$ will be used for conjunctions at type *e*: *John and Bill* will be interpreted as $j \vee b$, the sum of j and b. There is no problem with this if we structure *e* as a join semilattice. However, if *e* is a full Boolean algebra, then $\vee$ is only one of the operations on *e*. We have the other Boolean connectives available as well. That leads us to expect that we could interpret *John or Mary* as $j \wedge m$ and *Not Mary* as $\neg m$.

But that leads to interpreting *John or Mary* as the zero element, and making it equivalent with *Mary or Sue*. Similarly, *Not Mary* will be interpreted as 'everybody but Mary' (i.e. the sum of all atoms, except for Mary).

Clearly, *or* and *not* do not have such interpretations. Rather, what seems to be the case is that *and*, besides its interpretation in other domains as a Boolean connective can also at type *e* have the use of collecting individuals together into a plural element, and there are no corresponding interpretations for *or* and *not*. For this reason it is better to assume that *e* has just a join semilattice structure, and only *and* defined on it.

The third reason is conceptual. The arguments that lead us to adopt Boolean structures for other domains are typically arguments that concern the relation between the interpretation of the connectives at those domains. In this sense, those arguments are directly related to the axioms for Boolean algebras. We could argue axiom by axiom, why the connectives should satisfy them. The axioms that we get arise directly out of the conceptual needs that we have in characterizing the interpretations of related expressions like *and*, *or* and *not*.

This is not the case for the domain of individuals. We are led conceptually to the proper structures by thinking about individuation of plural

entities. We think about them as sums of individuals. We have a fundamental conceptual intuition that the sums of distinct sets of individuals should be distinct as well. This leads directly to free *i*-join semilattices, hence these structures are directly motivated by our conceptual needs. In fact it is fascinating to see how few conceptual assumptions we have to make to commit ourselves to these structures. It is a happy surprise that they can be interpreted as Boolean algebras, but that is not where we should look for their conceptual ontogenesis.

Let us turn now to the semantics.

The Language L

We will restrict ourselves to one place predicates (and one two place predicate). The language *L* has the following ingredients:

1. The standard first order operations $\neg$, $\wedge$, $\vee$, $\exists$ and abstraction λ.
2. Individual constants and individual variables.
3. Two term creating operations: $+$ and σ.
4. A special relational constant $\leqslant$.
5. A set **P** of one place predicates. This set is sorted into three different sets:

   IND: the set of individual level predicates

   COL: the set of collective predicates

   MIX: the set of mixed predicates

6. We have a special predicate $AT \in IND$
7. There are three predicate operations: $^{\uparrow}$, $^{\downarrow}$, ^{D}

Let me directly make a notational remark: $^{\uparrow}$ is written * in Link's work and related papers. I choose to use a different notation here because I pair it here with the down operation, (Another warning: these predicate operations are not related to the up and down term operations in Landman, 1989).

Syntax of L

TERM is the smallest set such that:

1. $CON \cup VAR \subseteq TERM$
2. if $t, t' \in TERM$, then $(t + t') \in TERM$

3. if $x \in VAR$ and $P \in PRED$, then $\sigma x.P(x) \in TERM$

PRED is the smallest set such that:

1. $\mathbf{P} \subseteq PRED$
2. If $P \in PRED$, then ${}^{\uparrow}P, {}^{\downarrow}P, {}^{D}P \in PRED$
3. if $x \in VAR$ and $\varphi \in FORM$, then $\lambda x.\varphi \in PRED$

FORM is the smallest set such that:

1. if $t \in TERM$ and $P \in PRED$, then $P(t) \in FORM$
2. if $\varphi, \psi \in FORM$ and $x \in VAR$, then $\neg\varphi$, $(\varphi \wedge \psi)$, $\exists x\varphi \in FORM$
3. if $t, t' \in TERM$, then $t \leqslant t' \in FORM$

Semantics for L

A *model* for L is a triple $\langle\langle A, \vee\rangle, *, i\rangle$ where:

1. $\langle A, \vee\rangle$ is a free i-join semilattice, generated by set of atoms AT. $PL = A - AT$
2. $* \notin A$ (this is the undefined element dealing with non-referring terms)
3. i is an interpretation function for the non-logical constants such that:
   1. if $c \in CON$, then $i(c) \in A \cup \{*\}$
   2. if $P \in IND$, then $i(P) \subseteq AT$
      if $P \in COL$, then $i(P) \subseteq PL$
      if $P \in MIX$, then $i(P) \subseteq A$

An assignment function is a function g: $VAR \to A$.
We define $[\![\ ]\!]_{M,g}$ (and suppress M):

TERM

1. $[\![c]\!]_g = i(c)$
2. $[\![x]\!]_g = g(x)$
3. $[\![t + t']\!]_g = [\![t]\!]_g \vee [\![t']\!]_g$ if both $[\![t]\!]_g, [\![t']\!]_g \in A$;
   $*$ otherwise.

4. $[\![\sigma x.P(x)]\!]_g = \vee([\![P]\!]_g)$ if $\vee([\![P]\!]_g) \in [\![P]\!]_g$; $*$ otherwise.

PRED

1. $[\![P]\!]_g = i(P)$, for $P \in \mathbf{P}$
   $[\![\leqslant]\!]_g = \leqslant$, where $x \leqslant y$ is defined as $x \vee y = y$
   $[\![AT]\!]_g = AT$
2. $[\![\lambda x.\varphi]\!]_g = \{d \in A: [\![\varphi]\!]_{gx}^{d} = 1\}$
3. $[\![{}^{\uparrow}P]\!]_g = [[\![P]\!]_g]$, the *sub* join semilattice generated by $[\![P]\!]_g$ under $\vee$ (note, this is not necessarily free).
4. $[\![{}^{\downarrow}P]\!]_g = \{d \in AT: d \in [\![P]\!]_g\}$
   ${}^{\downarrow}P$ could be defined as: $\lambda x.AT(x) \wedge P(x)$
5. $[\![{}^{D}P]\!]_g = \{d \in A: \forall a \in AT[$ if $a \leqslant d$ then $a \in [\![P]\!]_g]\}$
   ${}^{D}P$ could be defined as: $\lambda x.\forall y[AT(y) \wedge y \leqslant x \to P(y)]$

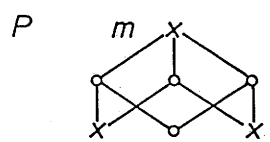
FORM

1. $[\![P(t)]\!]_g = 1$ iff $[\![t]\!]_g \in [\![P]\!]_g$; 0 otherwise
2. $[\![t \leqslant t']\!]_g = 1$ iff $[\![t]\!]_g \leqslant [\![t']\!]_g$; 0 otherwise
3. $[\![\neg\varphi]\!]_g = 1$ iff $[\![\varphi]\!]_g = 0$; 0 otherwise; etc.
4. $[\![\exists x\varphi]\!]_g = 1$ iff for some $d \in A$: $[\![\varphi]\!]_{gx}^{d} = 1$; 0 otherwise.

Let us show some aspects of these definitions.

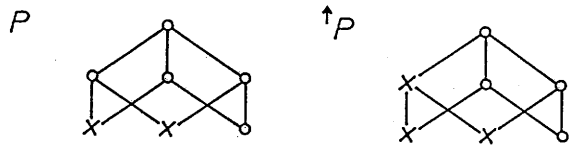
First concerning the definite article. $\sigma x.P(x)$ denotes $\vee[\![P]\!]$ if this is in $[\![P]\!]$, else it is undefined. This means that $\sigma x.P(x)$ is only defined if $[\![P]\!]$ forms itself an i-join semilattice, and then $\sigma x.P(x)$ is the maximal element of $[\![P]\!]$.

Note that it only means that $[\![P]\!]$ forms a join semilattice, *not* that it forms a sublattice of A. For instance if P is the mixed predicate indicated by the x-es in the following diagram, then $\sigma x.P(x)$ *is* defined, and is m, even though $[\![P]\!]$ is not a sublattice of A:



We are here only interested in the case where P is an individual level predicate. Preluding the discussion to come, we will assume that the

singular of an individual predicate like *Boy* is in *IND*, and that the plural of this predicate is $^{\uparrow}Boy$. So *Boy* denotes a set of atoms, and $^{\uparrow}Boy$ denotes the *i*-join semilattice generated by $[\![Boy]\!]$, for example:



Consequently, we will represent the singular definite *The boy* as $\sigma x.Boy(x)$ and the plural definite *The boys* as $\sigma x.{}^{\uparrow}Boy(x)$.

So we have a unified analysis of both the singular and the plural definite determiner. Moreover, the analysis has some nice consequences.

Since *Boy* is a set of atoms, it only forms a join semilattice if it consists of exactly one element, hence *The boy* is only defined if there is exactly one boy.

On the other hand, the plural predicate $^{\uparrow}Boy$ only *doesn't* form an *i*-join semilattice if the extension of *Boy* is empty, in all other cases $^{\uparrow}Boy$ denotes the join semilattice generated by *Boy*, and *the boys* denotes the maximal element of this set, which is that element that is the sum of all the boys.

Let us now turn to the analysis of plurality for predicates. Suppose we represent natural language predicate *K* as basic predicate *P* in our logical language. Then we have to associate with *K* both a representation for its singular and for its plural form.

I will make the assumption that if we associate *K* with *P*, then we identify either $SING(K)$ or $PLUR(K)$ with *P*. In particular, I will assume the following:

Let *K* be associated with basic predicate *P*.

$$\text{If } P \in IND \text{ then } SING(K) = P$$
$$\text{If } P \in COL \text{ then } PLUR(K) = P$$
$$\text{If } P \in MIX \text{ then } PLUR(K) = P$$

Secondly, I will assume that for each predicate, we have an operation of singularization, that singularizes a plural form, and an operation of pluralization that pluralizes a singular form.

Singularization is our operation $^{\downarrow}$

Pluralization is our operation $^{\uparrow}$

That is:

For every predicate *P*:

$$SING(P) = {}^{\downarrow}P$$
$$PLUR(P) = {}^{\uparrow}P$$

Finally, I will assume that the sets of all singular and plural representations of *K* are obtained by freely applying singularization and pluralization. This leads to the following:

$S_K = \{(SING(K)), SING(PLUR(K)), SING(PLUR\ (SING(K))), \ldots\}$
Here $(SING(K))$ indicates that this value is in the set if we have associated *K* with it.

$P_K = \{(PLUR(K)), PLUR(SING(K)), PLUR(SING\ (PLUR(K)), \ldots\}$

So we are assuming what seems to be quite some ambiguity. In fact, we do not get much ambiguity. Here is what we get:

We associate *K* with *P*.

If $P \in IND$ then:

$$S_K = \{P\}$$
$$P_K = \{{}^{\uparrow}P\}$$

Namely, we set $SING(K) = P$

$$PLUR(SING(K)) = {}^{\uparrow}P$$

all further singularizing and pluralizing switches you between these two, because clearly for individual level predicates:

$${}^{\downarrow}{}^{\uparrow}P = P$$

If $P \in COL$ then:

$$S_K = \emptyset$$
$$P_K = \{P, \emptyset\}$$

Namely, we set $PLUR(K) = P$. Since *P* is collective, it doesn't contain atoms, hence $SING(PLUR(K)) = \emptyset$. In this bivalid theory this means that it is undefined for atoms; hence there is no singular that can be meaningfully applied to singular individuals.

Of course, if we pluralize this, we do not get *P* back, rather we get

the empty set: $PLUR(SING(PLUR(K)) = \emptyset$. This means that we have a reading of the plural which cannot be meaningfully applied to anything. This will never show up, however, because in every case where this reading cannot be meaningfully applied, the other reading can *always* be meaningfully applied. That is, the theory claims that a sentence like *John and Bill meet* has a reading where this sentence is unwellformed. We do not experience this reading because it also has a reading where it *is* wellformed. Clearly, no other readings can be added after this.

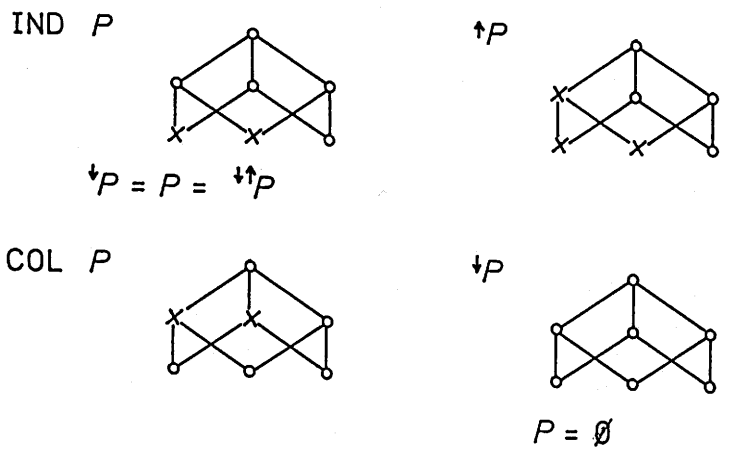
If $P \in MIX$ then:

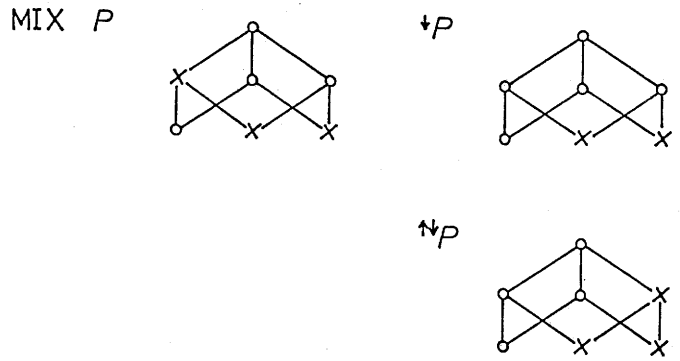
$$S_K = \{{}^{\downarrow}P\}$$
$$P_K = \{P, {}^{\uparrow}{}^{\downarrow}P\}$$

Namely, we have set $PLUR(K) = P$; $SING(PLUR(K)) = {}^{\downarrow}P$, denoting the set of atoms in P (in the case of *Carry a piano upstairs*, taken as a basic predicate this means that *Carries a piano upstairs* unambiguously denotes the set of singular individuals that all alone carry a piano upstairs).

We get an additional meaning from $PLUR(SING(PLUR(K)) = {}^{\uparrow}{}^{\downarrow}P$: (the join semilattice generated by the singular individuals that carry a piano upstairs). Hence the plural of mixed predicate K is ambiguous between P and ${}^{\uparrow}{}^{\downarrow}P$. (The singular is unambiguous, though: after this we get ${}^{\downarrow}P$ back.)

Let's look at this in an example. In the following diagram x again indicates that that particular element is in the extension of the relevant predicate.





Here are some consequences of these assumptions.

– Individual level predicates show cumulative reference and distributivity: these are resp. the left to right and the right to left part of the following pattern:

John is a boy and Bill is a boy iff John and Bill are Boys.

$$B(j) \wedge B(b) \leftrightarrow {}^{\uparrow}B(j + b)$$

Cumulative reference follows directly from the definition of ${}^{\uparrow}$. Distributivity follows from the fact that ${}^{\uparrow}B$ is freely generated by B, where B is a set of atoms: the only way $j \vee b$ can get to come into ${}^{\uparrow}B$ is by having both j and b in B.

Similar distributivity is seen for plural definites: suppose that the Nobel prize winners are among the scientists, then:

If the scientists are women then the Nobel prize winners are women.

$$\sigma x.{}^{\uparrow}N(x) \leqslant \sigma x.{}^{\uparrow}S(x) \wedge {}^{\uparrow}W(\sigma x.{}^{\uparrow}S(x)) \rightarrow {}^{\uparrow}W(\sigma x.{}^{\uparrow}N(x))$$

No such inference is found for collective predicates: *The boys and the girls meet* does not entail *The boys meet and the girls meet*, nor *vice versa* (but see Landman (1989) for discussion of distibutive readings of such collective predicates).

$$M(\sigma x.{}^{\uparrow}B(x) + \sigma x.{}^{\uparrow}G(x)) \quad \text{vs.}$$
$$M(\sigma x.{}^{\uparrow}B(X)) \wedge M(\sigma x.{}^{\uparrow}G(x))$$

The first sum can be in M without either one of the constituting second ones being in M, and *vice versa*.

Finally let's look at mixed predicates:

> *John carries a piano upstairs and Bill carries a piano upstairs iff John and Bill carry a piano upstairs.*

The first conjunction is represented as:

$$^{\downarrow}C(j) \wedge {}^{\downarrow}C(b)$$

The second sentence is ambiguous between: the collective $C(j + b)$ and the distributive $^{\uparrow\downarrow}C(j + b)$.

On the first reading, as for any collective predicate, the inference leftwards or rightwards is invalid, because there is no reason why $j + b$ should be in C if j and b are, nor *vice versa*.

On the second, distributive reading both cumulative reference and distributivity hold:

$$^{\downarrow}C(j) \wedge {}^{\downarrow}C(b) \leftrightarrow {}^{\uparrow\downarrow}C(j + b)$$

and for the same reason as for individual level predicates: $^{\downarrow}C$ is C restricted to its atoms. $^{\uparrow\downarrow}C$ is the free i-join semilattice generated by the latter set of atoms, clearly $j + b$ will be in this semi-lattice iff j and b are in this set of atoms.

Let us now look at some connections between our predicate operators. I already mentioned the first of the following:

FACT:

> *for all* $P \in IND$: $^{\uparrow\downarrow}P = P$
> *for all* $P \notin IND$: $^{\downarrow\uparrow}P = {}^{\downarrow}P$

Another fact that I have already mentioned is the following:

FACT: *If* $P \in IND$, *then* $[\![^{\uparrow}P]\!]$ *is the free i-join semilattice generated by* $[\![P]\!]$.

This fact follows from a more general theorem for free algebras:

THEOREM. *Let A be a free algebra generated by X, let* $Y \subseteq X$. *Then* $[Y]$ *is the free subalgebra of A generated by Y.*

Proof. Every function from Y into algebra B is the restriction $f \restriction Y$ of some function from X into B. We know that f can be extended to a homomorphism h from A into B. Look at $h \restriction [Y]$. Obviously $h \restriction [Y]$ extends $f \restriction Y$.

Furthermore $h \restriction [Y]$ is a homomorphism, because the restriction of a homomorphism from A into B to a subalgebra A' is a homomorphism from A' into B.

I have made the plural of mixed predicates ambiguous between P and $^{\uparrow\downarrow}P$. Link (in Link [forthcoming]) assumes that mixed predicates are ambiguous between P and ^{D}P. That leads us to the following:

THEOREM. *For every predicate* P: $^{\uparrow\downarrow}P = {}^{D}P$

Proof.

$$[\![^{D}P]\!] = \{x: \forall y[\text{if } y \in AT \text{ and } y \leqslant x \text{ then } y \in [\![P]\!]]\}$$
$$[\![^{\uparrow\downarrow}P]\!] = [\{y: y \in AT \text{ and } y \in [\![P]\!]\}]$$

Now:

1. $x \in [\![^{\uparrow\downarrow}P]\!]$ iff $x \in [\{y: y \in AT \text{ and } y \in [\![P]\!]\}]$
2. $x \in [\{y: y \in AT \text{ and } y \in [\![P]\!]\}]$ iff
   $AT_x \subseteq \{y: y \in AT \text{ and } y \in [\![P]\!]\}$

Here AT_x is the set of atoms below x. This we have proved in the previous chapter: x is the sum of atoms in P iff all the atoms lower than x are themselves atoms in P.

3. $AT_x \subseteq \{y: y \in AT \text{ and } y \in [\![P]\!]\}$ iff
   $x \in \{x: AT_x \subseteq \{y: y \in AT \text{ and } y \in [\![P]\!]\}\}$ iff
   $x \in \{x: \forall y[\text{if } y \in AT \text{ and } y \leqslant x \text{ then } y \in [\![P]\!]]\}$ iff
   $x \in [\![^{D}P]\!]$

Let me make one more algebraic connection. Suppose we want to give a functional theory, rather than the set theoretic one I gave above. Then predicates should be functions in $\langle A, t\rangle$. It is very easy, of course, to lift the semantics I gave to this domain: we take the old theory and let the predicates be the characteristic functions of the sets we gave there as the denotations of predicates:

Example. *If* $P \in IND$, then we can define: $i'(P) = CH(i(P))$. And what will hold is that for all $d \in PL$: $i'(P)(d) = 0$.

The domain of truth values, t is ordered as a complete join semilattice

under $\wedge$, $\langle\{0, 1\}, \wedge\rangle$. (That is, we look at this as a *join* semilattice, i.e. as ordered by $\geq$.)

Let us look at $^{\uparrow}$:

$$[\![^{\uparrow}P]\!]' = CH([\![P]\!])$$

THEOREM. *For every P:* $[\![^{\uparrow}P]\!]'$ *is the unique homomorphism in* $\langle A, t\rangle$ *that extends* $[\![^{\downarrow}P]\!]' \upharpoonright AT$.

Proof. We have to show:

1. $[\![^{\uparrow}P]\!]'$ extends $[\![^{\downarrow}P]\!]' \upharpoonright AT$.

This is obvious:

$$[\![^{\uparrow}P]\!]' = CH([\![P]\!]).$$

Let AT_P be the set of atoms in $[\![P]\!]$. $[\![^{\downarrow}P]\!]' \upharpoonright AT$ is the characteristic function of AT_P in $\langle AT, t\rangle$. It assigns 1 to all atoms in AT_P, 0 to all other atoms. But the elements of AT_P are all in $[\![^{\uparrow}P]\!]$, and no other atoms are, hence indeed $[\![^{\uparrow}P]\!]'$ extends $[\![^{\downarrow}P]\!]' \upharpoonright AT$.

2. $[\![^{\uparrow}P]\!]'$ is a homomorphism from $\langle A, \vee\rangle$ into $\langle\{0, 1\}, \wedge\rangle$.

$[\![^{\uparrow}P]\!]'(x \vee y) = 1$ iff $x \vee y \in [[\![P]\!]]$ iff $\vee(AT_{x\vee y}) \in [[\![P]\!]]$ iff $AT_{x\vee y} \subseteq AT_P$ iff $AT_x \cup AT_y \subseteq AT_P$ iff $AT_x \subseteq AT_P$ and $AT_y \subseteq AT_P$ iff $x \in [[\![P]\!]]$ and $y \in [[\![P]\!]]$ iff $CH([[\![P]\!]])(x) = 1$ and $CH([[\![P]\!]])(y) = 1$ iff $[\![^{\uparrow}P]\!]'(x) \wedge [\![^{\uparrow}P]\!]'(y) = 1$.

So indeed, $[\![^{\uparrow}P]\!]'$ is a homomorphism extending $[\![^{\downarrow}P]\!]' \upharpoonright AT$, and because A is free, this homomorphism is unique.

7.3. MASS NOUNS

In this section we will extend the theory of the previous section with mass nouns.

For the mass domain, we also want to be able to talk about *sums*, in order to represent terms like *The water in glass A and the water in glass B*.

However, we cannot simply take over the plural structures, because those are join semilattices generated by a set of minimal elements, and it is not clear that when we think about water and the parts of water, there are such minimal elements, i.e. that there are minimal parts of water.

Now, of course, we know that there are such minimal parts in the world, the water molecules. But there is good reason to assume that these physical minimal water particles do not play any role in the semantic domain of water and its parts.

Namely, assume that they do, i.e. assume that water is the set of all sums of water molecules. Now take the water in my cup. We know that it contains a finite number of water molecules, hence it contains a finite number of atoms. Say the number is 10 million. Since the water in my cup is the sum of these 10 million atoms, we would be conceptually able to count the water in my cup, hence we should be able to interpret

there is 10 million water(s) in my cup.

But, of course, we cannot. Water cannot be counted. We can measure how much water there is in my cup, we cannot count how much water there is.

This tells us, thus, that semantically, the structure of the mass domain should be a non-atomic structure, and maybe even an atomless structure. When we want to see what that structure should be, we should develop our conceptual intuitions concerning the notion part of in the mass domain.

These conceptual intuitions are tied up with the notion of *partition*. A fundamental intuition about a mass entity like the water in my bathtub, is that you can divide it into non-overlapping parts that together make up the water in the bathtub. Moreover, you can partition the water in the bathtub in countless different ways, yet put together again these parts make up the water in my bathtub. The fact that I stir it doesn't make a difference. However, if I add some water from my teacup to it, it is in an intensional sense still the water in my bathtub, but it is a different mass entity, it has different parts. Mass entities, thus, are determined by their parts.

Of course, there are intensional notions of part of, and they no doubt play an important semantic role as well, but the above conceptual intuition concerning the possibility of partitioning water in different ways, and yet getting the same water back leads to an extensional notion of mass entities: the identity criterion for a mass entity is determined solely by what parts it has (see Bunt, 1985). In this way, the partitioning idea leads us directly to *i*-join semilattices: the mass entity water is the sum of its parts.

However, just as we saw in the count domain that not every general join semilattice would do, the partition intuition here too leads to additional conceptual constraints.

In the first place, we have to make sure that there are *enough* parts: the partition intuition tells us that if I take some, but not all of the water out of the water of my bathtup, then there is something left which does not overlap with what I took out, because what I took out and what is left forms a partition of the water in my bathtub. There should also be enough parts in order to avoid unintuitive identifications. The situation is not different here from the case of the count domain: if I have three buckets of water A, B and C, then the water in A, B and C, respectively, have no parts in common. In that case, if I put the water in A and the water in B together, I would end up with a different body of water than if I were to put the water in A and the water in C together, and yet a different body of water were I to put the water in B and C together.

Secondly, if we define a partition of w as a maximal set of non-overlapping parts whose join is w, we have to make sure that we disallow structures where some set of elements satisfies this definition of partition, but intuitively doesn't.

Intuitively, if you look at a particular part A of a body of water and you make a partition of that body of water, you should find the parts of A back, distributed over the blocks of that partition: that is, either A is part of some block, or A overlaps some blocks. In the latter case, if you take the overlapping part in every block with which A overlaps, then those parts should form a partition of A.

The above requirements can be put in the form of the following two conditions:

Witness: if A is body of water and B is a *proper* part of A, then there has to be some other part of A that does not overlap B.

Distributivity: If you take a body of water and you take two parts A and B that together make up the whole body of water, then if you look at any part C of that body of water, it has to be either part of A, or part of B, or the sum of some part of A and some part of B.

We assume, thus that the mass domain has the form of a witnessed distributive join semilattice (for further applications of these structures, see Krifka, 1990). I will call them part-of structures.

First we define:

a *overlaps* b, $a \bigcirc b$ iff $\exists c$: $c \leq a$ and $c \leq b$

A *part-of* structure is an i-join semilattice $\langle A, \vee \rangle$ satisfying:

1. A does not have a minimum.
2. *distributivity*: if $a \leq b \vee c$ then $a \leq b$ or $a \leq c$ or $\exists b' \leq b \exists c' \leq c$: $a = b' \vee c'$
3. *witness*: if $a < b$ then $\exists c \leq b$: $\neg(a \bigcirc c)$

Let $B \subseteq A$.

X is a *blockset* in B iff X is a set of mutually non-overlapping elements of B.

We know, by the maximal chain principle that if X is a blockset in B then X can be extended to a maximal blockset in B.

LEMMA. *If X is a maximal blockset in $(a]$, $a \in A$ then* $\vee X = a$.

Proof. We know $\vee X \leq a$. Assume $a \not\leq \vee X$. Then we know by the witness postulate that there is some $z \leq a$ such that $\vee X$ and z don't overlap. But then $X \cup \{z\}$ would be a blockset in $(a]$ extending X and X would not be maximal. Hence $a \leq \vee X$.

So maximal blocksets are *partitions*.

Witness tells us that if $x < y$ then there is a z such that x and z don't overlap. Hence $\{x, z\}$ is a blockset. So we know that $\{x, z\}$ can be extended to a partition. This tells us that if an element a has a proper part, this proper part is one of the blocks in some partition of a.

THEOREM: *Every part-of structure can be turned into a complete Boolean algebra by adding a* 0. *Every complete Boolean algebra can be turned into a part-of structure by leaving out the* 0.

This theorem generalizes the theorem that we gave in Chapter Six for free i-join semilattices and complete atomic Boolean algebras. I will refer to the proof of that theorem in the following proof.

Proof. Let B be a complete i-Boolean algebra.
Let $A = \langle B - \{0\}, \vee \restriction A \rangle$.

We claim: A is a part-of structure.

We prove witness first: In B: let $x < b$ $(x \neq 0)$. We know that

$\neg x \wedge b \leq b$. $\neg x \wedge b$ is the *relative complement* of x in b, this means: $\neg x \wedge b$ is the unique element such that:

$$(\neg x \wedge b) \vee x = b; (\neg x \wedge b) \wedge x = 0$$

If $\neg x \wedge b = 0$, then $\neg x \wedge b$ would be the relative complement in b of both b and x, which is impossible, so $\neg x \wedge b \neq 0$, hence $\neg x \wedge b \in A$. But $x \wedge (\neg x \wedge b) = 0$, so in A x and $\neg x \wedge b$ do not overlap, hence witness is satisfied.

From this it directly follows that A doesn't have a minimum: there is no element lower than both x and $\neg x \wedge b$.

We prove distributivity. Let $x \leq y \vee z$ (in A). We prove distributivity in B:

$$x = x \wedge (y \vee z),$$

hence, by Boolean distributivity,

$$x = (x \wedge y) \vee (x \wedge z)$$

$x \wedge y \leq y \vee z$ and $x \wedge z \leq y \vee z$

So B satisfies distributivity. If $x \wedge y \neq 0$ and $x \wedge z \neq 0$ the argument carries over straightforwardly to A. If $x \wedge y = 0$ and $x \wedge z \neq 0$ what follows is that $x = x \wedge z$, so this also holds in A, hence $x \leq z$. Similarly in the other case $x \leq y$. It cannot be the case that both $x \wedge y$ and $x \wedge z$ are 0 (then x would be 0). Hence A satisfies distributivity.

So we have shown indeed that A is a part-of structure.

We show the other direction.

Let A be a part-of structure. We define $B = \langle A \cup \{0\}, \bigvee, \bigwedge, \neg, 0, \bigvee A\rangle$ exactly as we did before in the case of free i-join semilattices, with the exception that if $a \neq 0$, $\bigvee A$, we set: $\neg a = \{b \in A: \neg(b \circ a)\}$.

We claim that B is a complete i-Boolean algebra. The same arguments that we made in Chapter Six show that B is a complete i-lattice bounded by 0 and $\bigvee A$. Also B is clearly Boolean distributive: inspection of the proof in Chapter Six shows that this follows from distributivity: distributivity is precisely the property that we used there to show that B is Boolean distributive.

So we have to show that B is complemented.

1. a and $\bigvee\{x: \neg(x \circ a)\}$ don't overlap. Else some z would be part of a and not overlap a, so $a \wedge \neg a = 0$.
2. If $a \neq 0$, $\bigvee A$, we know, by witness, that there is an element in $\{x: \neg(x \circ a)\}$. If this set has only one element we have the four element Boolean algebra. If there is more than one element, then we know, by witness, that there are two elements smaller than $\neg a$ that don't overlap. Since this is a blockset in $(\neg a]$, we then know that there is a partition X of $\neg a$. Hence $\bigvee X = \neg a$.

We claim: $X \cup \{a\}$ is a partition of A. If $X \cup \{a\}$ were not a partition of A then some element z of A would not overlap any element in $X \cup \{a\}$. But then $z \leq \neg a$. Since z does not overlap any element in X, $X \cup \{z\}$ would be a partition of $\neg a$ extending X. But X was maximal, a partition. Hence indeed $X \cup \{a\}$ is a partition of A and hence $\bigvee(X \cup \{a\}) = \bigvee A$. But $\bigvee(X \cup \{a\}) = a \vee \bigvee X$, and $\bigvee X = \neg a$ because X is a partition, hence $a \vee \neg a = \bigvee A$. This completes the proof.

Here is a fact about complete Boolean algebras: If X is a subset of B such that for every x and y in X, $x \wedge y = 0$ then $[X]$, the i-join semilattice generated by X under $\bigvee$ is a free i-join semilattice (and hence isomorphic in the sense of the theorem to a complete atomic i-Boolean algebra). If X is such a subset in B then X is a blockset in the corresponding A, hence we have:

If A is a part-of structure and X is a blockset in A then $[X]$ is a free i-join semilattice.

So indeed, in a part-of structure, if we have three non-overlapping elements, their pairwise sums are distinct.

So much for the ontology. Let's now extend the semantics of plurals given in the last section to mass nouns.

Logic of Plurality and Mass Nouns

I will roughly follow Link (1983) here, though I will do some things a bit differently. For a general overview of mass nouns, see Pelletier and Schubert (1989).

For simplicity we will keep mass terms and count terms apart here.

We will have individual constants and variables over count entities as before, but add to this individual constants and variables over mass entities. We add a new set of mass predicate constants: *MASS*. We will use the symbols $+$, σ and $\leq$ ambiguously to denote the corresponding concepts either in the mass domain or the count domain. The context

makes it clear which one is meant. Further, we add some new notions: p, g, ^{P}, ^{G}, K. These notions will be introduced below.

Syntax of L

CTERM is the smallest set such that:

1. $CCON \cup CVAR \subseteq CTERM$
2. if $t, t' \in CTERM$, then $(t + t') \in CTERM$
3. if $x \in CVAR$ and $P \in CPRED$, then $\sigma x.P(x) \in CTERM$
4. if $t \in MTERM$ then $p(t) \in CTERM$

MTERM is the smallest set such that:

1. $MCON \cup MVAR \subseteq MTERM$
2. if $t, t' \in MTERM$, then $(t + t') \in MTERM$
3. if $x \in MVAR$ and $P \in MPRED$, then $\sigma x.P(x) \in MTERM$
4. if $t \in CTERM$ then $g(t) \in MTERM$

CPRED is the smallest set such that:

1. $\mathbf{P} \subseteq CPRED$
2. If $P \in CPRED$, then ${}^{\uparrow}P$, ${}^{\downarrow}P$, ${}^{D}P \in CPRED$
3. if $x \in CVAR$ and $\varphi \in FORM$, then $\lambda x.\varphi \in CPRED$
4. if $P \in MPRED$ then ${}^{P}P \in CPRED$

MPRED is the smallest set such that:

1. $MASS \subseteq CPRED$
2. If $x \in MVAR$ and $\varphi \in FORM$, then $\lambda x.\varphi \in MPRED$
3. if $P \in CPRED$ then ${}^{G}P \in MPRED$

FORM is the smallest set such that:

1. if $t \in CTERM$ and $P \in CPRED$, then $P(t) \in FORM$
   if $t \in MTERM$ and $P \in MPRED$, then $P(t) \in FORM$
2. if $\varphi, \psi \in FORM$ and $x \in CVAR$, $y \in MVAR$, then $\neg\varphi$, $(\varphi \wedge \psi)$, $\exists x\varphi$, $\exists y\varphi \in FORM$
3. if $t, t' \in CTERM$, then $t \leqslant t' \in FORM$
   if $t, t' \in MTERM$, then $t \leqslant t' \in FORM$

4. if $t \in MTERM$ and $t' \in CTERM$ then $tKt' \in FORM$

Semantics of L

A *model* for L is a tuple $M = \langle\langle C, \vee, AT\rangle, \langle M, \vee\rangle, K, p, g, i, *\rangle$ where:

1. $\langle C, \vee, AT\rangle$ is an i-join semilattice freely generated by AT: *the count domain* of singular and plural entities.
2. $\langle M, \vee\rangle$ is a part-of structure: the *mass domain*.
3. $K \subseteq M \times C$, the relation of *material part of*.

This relation has to satisfy the following condition:

For every $c \in C$: $\{x \in M: xKc\}$ is a part-of structure, in particular:

$$\{x \in M: xKc\} = [\cup_{a \in AT \,\&\, a \leqslant c}\{y \in M: yKa\}]$$

This means the following. Consider a plural entity c. To get all the material parts of c, you have to look at the atomic parts of c. For every such atom, its material parts are required to form a part-of structure. Take the set of all the material parts of the atoms of c. The structure generated under sum by that set is a part-of structure: it is the set of all material parts of c.

4. the *grinder* function g is that function $g: C \to M$ such that for every $c \in C$: $g(c) = \vee\{x \in M: xKc\}$

So g maps an individual onto the sum of its material parts. g can also be read to stand for 'gruesome', because it is used to deal with the use of 'sie' in the last line of Brecht's 'Vom ertrunkenen Mädchen':

Dann ward sie Aas in Flüssen mit vielem Aas.

5. the *packaging* function p is a function $p: M \to AT$ such that for every $m \in M$: $g(p(m)) = m$

The packaging function allows us to use a mass entity as an atomic individual. Note that the above clause does not commit us to the materialistic view that an individual is nothing but the sum of its material parts. It does not say that $p(g(j)) = j$ $(j \in C)$.

6. i, the interpretation function, interprets the count constants as count entities, the mass constants as mass entities, it

interprets every count predicate as before. If $P \in MASS$ then for some $m \in M$: $i(P) = (m]$

I.e. $i(P)$ is the set of all parts of some mass entity. This is a sub-part-of structure of M.

Assignment function g assigns a mass entity to every mass variable and a count entity to every count variable.

I will only give the semantic interpretation for the new operations and relations here. $+$, σ and $\leqslant$ get exactly the same semantic interpretation in both the count domain and the mass domain, that is: if $+$ relates count terms it denotes sum in the count domain, if it relates mass terms it denotes sum in the mass domain, etc. Abstraction and quantification are sorted: if x is a count variable $\lambda x.\varphi$ is the set of count entities that have φ and $\exists x\varphi$ says that some count entity has φ. Similarly, if x is a mass variable. All clauses get their obvious interpretation. We add:

1. if $t \in CTERM$ then $[\![g(t)]\!]_g = g([\![t]\!]_g)$
2. if $t \in MTERM$ then $[\![p(t)]\!]_g = p([\![t]\!]_g)$
3. if $t \in MTERM$, $t' \in CTERM$ then $[\![tKt']\!]_g = 1$ iff $[\![t]\!]_g K [\![t']\!]_g$; 0 otherwise.
4. if $P \in MPRED$ then $[\![^P P]\!]_g = \{\vec{p}(m): m \in [\![P]\!]_g\}$
5. if $P \in CPRED$ then $[\![^G P]\!]_g = \{m \in M: mK\vee([\![P]\!]_g)\}$

This allows us to use a mass predicate as a singular count predicate of its packaged elements and similarly a count predicate as a mass predicate by grinding the sum of its elements and adding all the parts of that grind. This is what happens in the Brecht example: ${}^G(Aas)(g(sie))$.

Some facts:

If $P \in$ MASS then P is both distributive and satisfies cumulative reference:

$$P(t + t') \leftrightarrow P(t) \wedge P(t')$$

in fact: $P(t) \leftrightarrow \forall x[x \leqslant t \rightarrow P(x)]$.
This follows from the fact that P denotes a part-of structure.

If $x \in [\![P]\!]$ then all its parts are in $[\![P]\!]$, by definition of $[\![P]\!]$. If all its parts are in $[\![P]\!]$, then $x \in [\![P]\!]$, because x is the sum of its parts and $[\![P]\!]$ is closed under sum.

This means that we have the following equivalence:

The liquid in glass A and the liquid in glass B is whisky.

iff

The liquid in glass A is whisky and the liquid in glass B is whisky.

If we represent *liquid in glass A*, *liquid in glass B* and *whisky* resp. by the mass predicates LA, LB and W:

$$W(\sigma x.LA(x) + \sigma x.LB(x)) \leftrightarrow$$
$$W(\sigma x.LA(x)) \wedge W(\sigma x.LB(x))$$

We can deal with the puzzle of material parts, the fact that the following is consistent:

This ring is new but the gold in this ring is not new.

We can assume that *new* applies both to mass and count entities, (i.e. we have $N \in IND$ and a homophonic $N \in MASS$). For *gold* we have $G \in MASS$, and we have count constant r denoting this ring. Then we can represent the above example as:

$$N(r) \wedge \neg N(\sigma x.G(x) \wedge xKr)$$

This is perfectly consistent. There is no connection between individual r being new and the sum of its material parts that are gold being new.

Another fact: If $P \in MASS$ then $P(t) \leftrightarrow {}^P P(p(t))$. This is obvious from the definition of ${}^P P$.

Mass predicates, we have seen above, are treated as inherently distributive. This seems to work fine for predicates like *be whisky*. If the liquid in my glass is whisky, it is not counterintuitive to assume that its parts are whisky as well. For other predicates this may be more problematic, and for yet others this seems plainly wrong.

If the liquid in my glass is red, does that mean that every part of it is red? This may still seem plausible. What about dirty? If the liquid in my glass is dirty, does that mean that every part is dirty? What if the dirt is just floating on top? Or take: the liquid in my glass smells dirty? Is that false if some part of it doesn't smell dirty, say, doesn't smell at all? Clearly non-distributive is the following:

The water in the North Sea carries a ship from England to Holland.

That surely does not mean that all the water carries a ship from England to Holland.

Similarly non-distributive is:

> *The water in the North Sea and the water in the IJssel Lake is separated by a dike.*

So we seem to have to recognize non-distributive predicates in the mass domain as well. We could do that by assuming predicates whose extension does not not form a part-of structure. However that leads to problems (the very same argument that I will make here applies to the treatment of collective predicates as well and led to the revised theory of plurality that I proposed in Landman, 1989).

Assume that *smells bad* is such a non-distributive predicate: In *The water in A smells bad*, *smells bad* does not distribute. Yet it seems that the following equivalence is completely unproblematic:

(1) *The water in A and the water in B smells bad.*
$SB(\sigma x.WA(x) + \sigma x.WB(x))$

iff

(2) *The water in A smells bad and the water in B smells bad.*
$SB(\sigma x.WA(x)) \wedge SB(\sigma x.WB(x))$

This is only possible if in (1) the predicate *SB* distributes to $\sigma x.WA(x)$ and $\sigma x.WB(x)$. The problem is that you cannot make this predicate distribute from $\sigma x.WA(x) + \sigma x.WB(x)$ to these parts without it distributing to the rest as well. $\sigma x.WA(x)$ and $\sigma x.WB(x)$ denote mass entities that are in no way different from other mass entities. For instance, saying that *SB* in (1) only distributes to its direct parts would not help: take a non-smelling part of the water in A: a; take the rest of the water in A and sum it with the water in B: b; $\sigma x.WA.(x) + \sigma x.WB.(x) = a + b$, so *SB* would distribute to b; but b didn't smell bad.

The problem thus is: how to get the distributivity in (1), while keeping our predicates from distributing all the way down?

We can solve this problem with help of our packaging function p. p lets us think of a mass entity, stuff, as a count entity. We can assume that mass terms, like *the water in A* can shift their interpretation naturally from the mass domain to the count domain: when we predicate a mass property of it, we think of it as a mass entity; but when we predicate properties like *smells bad* or *carries a ship*, we predicate these

properties to them as count entities: we interpret *the water in A* as the body of water in A, taken as a whole. It is the body of water in the North Sea that, taken as an entity in itself, carries the ship. Asking which mass parts of this entity actually do any carrying is semantically as irrelevant as asking which parts of John do any carrying, when John carries a book. Similarly, if we can shift *the water in A* to the count domain, we shouldn't expect *smells bad* to distribute to all the parts, just as we don't expect it to distribute to all John's parts if John smells bad.

Let's assume then that (2) is represented as:

(3) $SB\ (p(\sigma x.WA(x)) \wedge SB\ (p(\sigma x.WB(x))$

where *SB* is a *count predicate* in *IND*.

Let's assume further that the singular morphology on the *verb* in (1) is not a semantic indicator, but is triggered by the fact that the *nouns* in the conjunctive subject with which it agrees are mass nouns.

Given this, we can represent (1) as we would represent it if the terms were count terms:

(4) $^{\uparrow}SB(p(\sigma x.WA(x)) + p(\sigma x.WB(x)))$

which is indeed equivalent to (3).

Hence, by using packaging, that is, shifting from mass terms to count terms, we can indeed insure that the predicate in (1) distributes to its parts as in (2), but not further down.

Packaging is essential also for other reasons in the earlier example:

> *The water in the North Sea and the water in the IJssel Lake is separated by a dike.*

Making *is separated by a dike* a collective mass predicate, i.e. a predicate that applies collectively to $\sigma x.WN(x) + \sigma x.WIJ(x)$ would give wrong results: let $\sigma x.A(x)$ be the sum of the water in the north part of the North Sea and the north part of the IJssel Lake and $\sigma x.B(x)$ the sum of the water in the south part of each. Then, since $\sigma x.WN(x) + \sigma x.WIJ(x) = \sigma x.A(x) + \sigma x.B(x)$, $SEP(\sigma x.WN(x) + \sigma x.WIJ(x))$ would entail $SEP(\sigma x.A(x) + \sigma x.B(x))$.

If we assume, however that in this example *SEP* applies collectively in the count domain to the sum of packages:

$$SEP(p(\sigma x.WN(x)) + p(\sigma x.WIJ(x)))$$

we avoid this problem.

ANSWERS TO EXERCISES

CHAPTER ONE

Exercise 1

(a) If φ is not true in M, then for some g: $M \not\models \varphi$. Then (by the lemma) for all g: $M \not\models \varphi$, hence φ is false in M. Similarly, if φ in not false in M, by the same argument φ is true in M. Hence φ cannot be both not true in M and not false in M. So φ is either true or false in M.

(b) We know that for some $d \in D$: $d \in i(P)$ and for some $d' \in D$: $d' \notin i(P)$. Hence we know that for some g (a g where $g(x) = d$) $M \models P(x)[g]$ and that for some g' (a g' where $g'(x) = d'$) $M \not\models P(x)[g]$. The first tells us that $P(x)$ is not true, the second that $P(x)$ is not false, so $P(x)$ is undefined. Of course, the same argument applies to $\neg P(x)$.

(c) Take any assignment g. $M \models P(x) \vee \neg P(x)[g]$ iff $M \models P(x)[g]$ or $M \models \neg P(x)[g]$ iff $g(x) \in i(P)$ or $g(x) \notin i(P)$. The latter is of course true (for any g). So for all g: $M \models P(x) \vee \neg P(x)[g]$, so $P(x) \vee \neg P(x)$ is true in M (in any M, for that matter). A similar argument tells you that $P(x) \wedge \neg P(x)$ is false in M, because there is no g such that $g(x) \in i(P)$ and $g(x) \notin i(P)$.

So you see that, although $P(x)$ and $\neg P(x)$ themselves are undefined, $P(x) \vee \neg P(x)$ and $P(x) \wedge \neg P(x)$ are not.

Exercise 2

$\varphi[x]$ and $\forall x\varphi[x]$ are logically equivalent if for every M: $M \models \varphi[x]$ iff $M \models \forall x\varphi[x]$. $M \models \varphi[x]$ iff for every g: $M \models \varphi[x][g]$ iff for every g and every $d \in D$: $M \models \varphi[x][g_x^d]$ if for every g: $M \models \forall x\varphi[x][g]$ iff $M \models \forall x\varphi[x]$.

Exercise 3

1. Every atomic formula has an equal number of left and right brackets.
2. Assume φ has an equal number of left and right brackets. $\neg\varphi$