7 Conjunctions, Plurals, Mass Terms, and Kinds

In this section we will be concerned with various semantic phenomena that appear when we consider plural noun phrases, conjunctions, and mass terms. We will see that they will lead us to a quite fundamental reconsideration of the way how certain NPs work.

7.1NP Conjunctions and Their Interpretation

7.1.1 VP Coordination

In chapter 3 we have discussed a way of extending the syntax and semantic interpretation of negation, which is basically a sentence operator (type D_{tt}), to an operator that is of type $D_{(et)et}$, that is, a VP operator. There is evidence that we should do the same for conjunction and disjunction, and we have done so in various homeworks.

Let us repeat that here. First, we know that *and* and *or* can conjoin sentences, and hence should be of type D_{ttt}:

(1) a. $[_S[_S$Leopold is asleep] and $[_S$Molly is awake]].
b. $[_S[_S$Leopold is asleep] or $[_S$Molly is asleep]].

But if the subject is the same, we rather find that *and* and *or* conjoin VPs directly:

(2) a. Leopold $[_{VP}$ $[_{VP}$slept] and $[_{VP}$snored]].
b. Molly $[_{VP}[_{VP}$loves Leopold] or $[_{VP}$knows Stephen]].

The alternatives, e.g. *Leopold slept and Leopold snored*, sound rather pedantic. So we should assume that *and* and *or* actually can be used according to the following syntactic rules:

(3) VP → VP Conj VP

Now, the meanings for *and* and *or* that we have specified in chapter 3, repeated here, certainly does not fit this rule.

(4) a. $[\![$*and*$]\!] = \lambda t \quad D_t[\lambda t' \quad D_t[\textsc{min}(\{t, t'\})]]$
b. $[\![$*or*$]\!] = \lambda t \quad D_t[\lambda t' \quad D_t[\textsc{max}(\{t, t'\})]]$

These meanings required sentences as arguments, but now we should require VP-meanings. We would like to say that, for example, *and* is ambiguous between (4.a) and (5):

(5) $[\![$*and*$]\!]$, as VP conjunction: $\lambda P \quad D_{et}[\lambda Q \quad D_{et}[\lambda x \quad D_e[\textsc{min}(\{P(x), Q(x)\})]]]$

Notice that the VP-conjunction meaning takes two VP meanings, represented by the placeholder variables P and Q, and gives a VP meaning specified by the lambda term $\lambda x \quad D_e[...]$. It gives us the value true if both P and Q, when applied to x, give us the value true, and false otherwise. This appears to be the right interpretation.

If we interpret *and* as in (5.b) we get derivations like the following:

(6) a. ⟦[$_S$ [$_{NP}$ *Leopold*] [$_{VP}$ [$_{VP}$ [$_V$ *sleeps*]] [$_{Conj}$ *and*] [$_{VP}$ [$_V$ *snores*]]]]⟧
b. = ⟦[$_{VP}$ [$_{VP}$ [$_V$ *sleeps*]] [$_{Conj}$ *and*] [$_{VP}$ [$_V$ *snores*]]]]⟧(⟦[$_{NP}$ *Leopold*]⟧)
c. = ⟦*and*⟧(⟦*sleeps*⟧)(⟦*snores*⟧)(⟦*Leopold*⟧)
d. = λP D_{et}[λQ D_{et}[λx D_e[MIN({P(x), Q(x)})]]](λy D_e[y sleeps])(λz D_e[z snores])(LB)
e. = λx D_e[MIN({λy D_e[y sleeps](x), λz D_e[z snores](x)})](LB)
f. = λx D_e[MIN([x sleeps], [x snores])](LB)
g. = MIN([LB sleeps], [LB snores])
h. = 1, if [LB sleeps] = [LB snores] = 1,
= 0, else.

We get the same result as we would have gotten for *Leopold sleeps and Leopold snores.*

The meaning of *or* can be given in a similar way:

(7) ⟦*or*⟧ as VP conjunction: λP D_{et}[λQ D_{et}[λx D_e[MAX({P(x), Q(x)})]]]

There is an important point to be made here. We just have assumed that the meanings of *and* and *or* are actually **ambiguous**. Granted that their ambiguity is systematic, quite unlike to the erratic ambiguity of words like *pen* or *bank*. But they are ambiguous. Now, so far we have given the meaning of an expression α by rules of the format ⟦α⟧ = ..., that is, we gave a definite meaning to an expression α. But now we have a situation in which α can have one of two meanings. One way is to make a difference between different kinds of α's. For example, we can differentiate between *and* and *and′*, and have the following intepretation rules:

(8) a. ⟦*and*⟧ = λt D_t[λt′ D_t[MIN({t, t′})]
b. ⟦*and′*⟧ = λP D_{et}[λQ D_{et}[λx D_e[MIN({P(x), Q(x)})]]]

We will later see that we can avoid stipulating such spurious ambiguities.

We have seen that the interpretation of conjunction and disjunction can be systematically ambiguous. The meaning of *and* for sentences and for verb phrases can be given on the basis of the function MIN. The only difference is in the type of arguments that *and* expects, and the type of value that it delivers. We therefore say that these expressions show **type flexibility**. Notice that our philosophy, to assume fairly general semantic rules that combine meanings in whatever way possible, pays off here: The overall system of our grammar is quite simple, we just have to assume systematic ambiguity for certain expressions.

7.1.2 Coordination for Other Types of Expressions

The conjunction *and* and the disjunction *or* can be used to conjoin other types of expressions as well. For example, we can conjoin quantifiers:

(9) a. [$_{NP}$ [$_{NP}$ A girl] and [$_{NP}$ every boy]] came.
b. [$_{NP}$ [$_{NP}$ Seven apples] or [$_{NP}$ five pears]] were in the bowl.

Now, quantifiers are of type $D_{(et)t}$, and we can define a meaning for conjunction and disjunction for this type, as follows:

(10) a. ⟦*and*⟧, as quantifier conjunction: λQ $D_{(et)t}$λQ′ $D_{(et)t}$λP D_{et}[MIN{Q(P), Q′(P)}]
b. ⟦*or*⟧, as quantifier disjunction: λQ $D_{(et)t}$λQ′ $D_{(et)t}$λP D_{et}[MAX{Q(P), Q′(P)}]

Consider the following derivation of (9.a):

(11)a. $\llbracket[_S\ [_{NP}\ [_{NP}\ \textit{a girl}]\ \textit{and}\ [_{NP}\ \textit{every boy}]]\ [_{VP}\textit{came}]]\rrbracket$

b. $= \llbracket[_{NP}\ [_{NP}\ \textit{a girl}]\ \textit{and}\ [_{NP}\ \textit{every boy}]]\rrbracket(\llbracket \textit{came}\rrbracket)$

c. $= \llbracket \textit{and}\rrbracket(\llbracket[_{NP}\ \textit{a girl}]\rrbracket)(\llbracket[_{NP}\ \textit{every boy}]\rrbracket(\llbracket \textit{came}\rrbracket)$

d. $= \lambda Q\ \ \ D_{(et)t}\lambda Q'\ \ \ D_{(et)t}\lambda P\ \ \ D_{et}[\textsc{min}\{Q(P), Q'(P)\}]$
$(\lambda P'\ \ \ D_{et}[\llbracket girl\rrbracket \cap P' \neq\ \ \ ])(\lambda P''\ \ \ D_{et}[\llbracket boy\rrbracket\ \ \ \ P''])(\llbracket came\rrbracket)$

e. $= \lambda P\ \ \ D_{et}[\textsc{min}\{\lambda P'\ \ \ D_{et}[\llbracket girl\rrbracket \cap P' \neq\ \ \ ](P), \lambda P''\ \ \ D_{et}[\llbracket boy\rrbracket\ \ \ \ P''](P)\}]((\llbracket came\rrbracket)$

f. $= \textsc{min}\{\lambda P'\ \ \ D_{et}[\llbracket girl\rrbracket \cap P' \neq\ \ \ ](\llbracket came\rrbracket), \lambda P''\ \ \ D_{et}[\llbracket boy\rrbracket\ \ \ \ P''](\llbracket came\rrbracket)\}$

g. $= \textsc{min}\{[\llbracket girl\rrbracket \cap \llbracket came\rrbracket \neq\ \ \ ], [\llbracket boy\rrbracket\ \ \ \ \llbracket came\rrbracket]\}$

h. $= 1$, if $[\llbracket girl\rrbracket \cap \llbracket came\rrbracket \neq\ \ \ ] = [\llbracket boy\rrbracket\ \ \ \ \llbracket came\rrbracket] = 1$,
$= 0$, else.

We get the truth value 1 if both the sentences *a girl came* and *every boy came* are true, and else 0. This is obviously the right result. The meaning rule for *and* given in (9.a) gives us a way of reducing quantifier conjunction to sentence conjunction.

We can go even further. For we find cases in which determiners are coordinated:

(12)a. $[_{NP}\ [_{Det}\ [_{Det}$ more than three] and/but $[_{Det}$ less than seven]] boys] came.

b. $[_{NP}\ [_{Det}\ [_{Det}$ exactly two] or $[_{Det}$ exactly five]] boys] came.

We can handle such cases by assuming that *and* and *or* have meanings that allow to combine determiner meanings, which are of type $D_{(et)(et)t}$. Here is the rule for conjunction:

(13) $\llbracket \textit{and/but}\rrbracket$, as determiner conjunction:
$\lambda D\ \ \ D_{(et)(et)t}\lambda D'\ \ \ D_{(et)(et)t}\lambda P\ \ \ D_{et}\lambda P'\ \ \ D_{et}[\textsc{min}\{D(P)(P'), D'(P)(P')\}]$

We now can derive example (12.a), as follows:

(14)a. $\llbracket[_S[_{NP}\ [_{Det}\ [_{Det}\ \textit{more than three}]\ \textit{but}\ [_{Det}\ \textit{less than seven}]]\ \textit{boys}]\ [_{VP}\textit{came}]]\rrbracket$

b. $= \llbracket[_{NP}\ [_{Det}\ [_{Det}\ \textit{more than three}]\ \textit{but}\ [_{Det}\ \textit{less than seven}]]\ \textit{boys}]\rrbracket(\llbracket came\rrbracket)$

c. $= \llbracket[_{Det}\ [_{Det}\ \textit{more than three}]\ \textit{but}\ [_{Det}\ \textit{less than seven}]]\rrbracket(\llbracket boys\rrbracket)(\llbracket came\rrbracket)$

d. $= \llbracket but\rrbracket(\llbracket[_{Det}\ \textit{more than three}]\rrbracket)(\llbracket[_{Det}\ \textit{less than seven}]\rrbracket)(\llbracket boys\rrbracket)(\llbracket came\rrbracket)$

e. $= \lambda D\ \ \ D_{(et)(et)t}\lambda D'\ \ \ D_{(et)(et)t}\lambda P\ \ \ D_{et}\lambda P'\ \ \ D_{et}[\textsc{min}\{D(P)(P'), D'(P)(P')\}]$
$(\llbracket[_{Det}\ \textit{more than three}]\rrbracket)(\llbracket[_{Det}\ \textit{less than seven}]\rrbracket)(\llbracket boys\rrbracket)(\llbracket came\rrbracket)$

f. $= \textsc{min}\{\llbracket[_{Det}\ \textit{more than three}]\rrbracket(\llbracket boys\rrbracket)(\llbracket came\rrbracket), \llbracket[_{Det}\textit{less than seven}]\rrbracket(\llbracket boys\rrbracket)(\llbracket came\rrbracket)\}$

g. $= \textsc{min}\{[\#(\llbracket boy\rrbracket \cap \llbracket came\rrbracket) \geq 3], [\#(\llbracket boy\rrbracket \cap \llbracket came\rrbracket) \leq 7]\}$

h. $= 1$, if $\#(\llbracket boy\rrbracket \cap \llbracket came\rrbracket) \geq 3]$ and $\#(\llbracket boy\rrbracket \cap \llbracket came\rrbracket) \leq 7$,
$= 0$, else.

We get the same result as for the sentence More than three boys came and/but less than seven boys came.

7.1.3 *Conjunction and Disjunction Generalized*

We have seen that conjunction and disjunction, which are originally defined for meanings of type t, can be extended to meanings of a variety of other types. We have discussed, in particular, extensions to meanings of type et, of type (et)t, and of type (et)(et)t. It is relatively easy to define what *and* and *or* should mean for other constituents, like transitive verbs (type eet) or attributive adjectives (type (et)et):

(15)a. Leopold knows and likes Stephen.
b. a long and difficult book

The question is, why is this now relatively easy? What are the principles that allow us to give the meanings of *and* and *or* for different types?

One important thing to notice is that all our definitions of *and* and *or* boil down to the use of MIN and MAX, which basically can be applied to a set of expressions of type t. This shows up in the fact that all the meanings for which we can define disjunctions and conjunctions “end” in the type t — we have defined it for types e**t**, ee**t**, (et)**t**, (et)e**t**, (et)(et)**t**, etc.

Secondly, we formulated conjunction and disjunction for various types by abstracting over the remaining arguments. For example, for the conjunction for type et we abstracted over the e argument, for the conjunction of type (et)t we abstracted over the et argument, and for conjunction for type (et)(et)t we abstracted over the two (et) arguments. Consider this case again (cf. (13)); P and P′ are the two arguments of type et that we have to abstract over.

(16) $\lambda D \quad D_{(et)(et)t}\lambda D' \quad D_{(et)(et)t}\lambda P \quad D_{et}\lambda P' \quad D_{et}[\text{MIN}\{D(P)(P'), D'(P)(P')\}]$

This allows us now to define conjunction in general as follows:

(17) ⟦*and*⟧ is defined as follows:
a. If α, β are meanings of type t, then ⟦*and*⟧$(\alpha)(\beta) = \text{MIN}\{\alpha, \beta\}$
b. If α, β are of type $(\sigma)\tau$, then ⟦*and*⟧$(\alpha)(\beta) = \lambda X \quad D_\sigma[$⟦*and*⟧$(\alpha(X))(\beta(X))]$

This is a recursive definition. The first clause (a) gives the basic case, if the arguments are both of type t. The second clause reduces the meaning of *and* for arguments of type $(\sigma)\tau$ to the simpler case of arguments of type τ. The type τ might be complex, and things would have to be reduced even further.

Notice that ⟦*and*⟧, according to this definition, is a function that allows for arguments of many different types. If we want to describe its type, we would have to say that it is of type $(\tau)(\tau)\tau$, where τ can be any type that ends in t. We would have to extend our notion of types accordingly, essentially to allow that meanings belong to SETS of types.

Let us see how definiton (17) works in two cases:

(18)a. Assume α, β are of type et, then ⟦*and*⟧$(\alpha)(\beta) = \lambda x \quad D_e[$⟦*and*⟧$(\alpha(x))(\beta(x))]$,
as $\alpha(x), \beta(x)$ are of type t, ⟦*and*⟧$(\alpha(x))(\beta(x)) = \text{MIN}\{\alpha(x), \beta(x)\}$,
hence ⟦*and*⟧$(\alpha)(\beta) = \lambda x \quad D_e[\text{MIN}\{\alpha(x), \beta(x)\}]$

b. Assume α, β are of type eet, then ⟦*and*⟧$(\alpha, \beta) = \lambda x \quad D_e[$⟦*and*⟧$(\alpha(x))(\beta(x))$,
as $\alpha(x), \beta(x)$ are of type et, ⟦*and*⟧$(\alpha(x), \beta(x)) = \lambda y \quad D_e$⟦*and*⟧$(\alpha(x)(y))(\beta(x)(y))$,
as $\alpha(x)(y), \beta(x)(y)$ are of type t, ⟦*and*⟧$(\alpha(x)(y))(\beta(x)(y)) = \text{MIN}\{\alpha(x)(y), \beta(x)(y)\}$
hence ⟦*and*⟧$(\alpha, \beta) = \lambda y \quad D_e\lambda x \quad D_e[\text{MIN}\{\alpha(x)(y), \beta(x)(y)\}]$

Notice that in the case of (b), the meaning of *and* for type eet is deduced in two steps. First, it is reduced to the meaning of *and* for type et, and this in turn is reduced to the meaning of *and* for type t. This is, as we know, typical for recursive definitions.

7.2 Collective Predication

7.2.1 Conjunction for Names

We have learned how to handle conjunction and disjunction for quantifiers. But of course it is possible to coordinate simple names with *and* or *or*:

(19) a. Leopold and Molly are asleep.
b. Leopold or Molly are awake.

We have seen that names, though basically of type e, can be lifted to the type of quantifiers, (et)t, which then allows us to conjoin them:

(20) a. ⟦*Leopold*⟧, as a quantifier: $\lambda P \quad D_{et}$[P(LB)]
b. ⟦*Molly*⟧, as a quantifier: $\lambda P \quad D_{et}$[P(MB)]
c. ⟦*Leopold and Molly*⟧
= ⟦*and*⟧(⟦*Leopold*⟧)(⟦*Molly*⟧)
= $\lambda P \quad D_{et}$[MIN{P(⟦*Leopold*⟧), P(⟦*Molly*⟧)}]
d. ⟦[$_S$ [$_{NP}$ *Leopold and Molly*] [$_{VP}$ *are asleep*]]⟧
= $\lambda P \quad D_{et}$[MIN{P(⟦*Leopold*⟧), P(⟦*Molly*⟧)}](⟦*asleep*⟧)
= MIN{⟦*asleep*⟧(⟦*Leopold*⟧), ⟦*asleep*⟧(⟦*Molly*⟧)}
e. = 1, of ⟦*asleep*⟧(⟦*Leopold*⟧) = ⟦*asleep*⟧(⟦*Molly*⟧) = 1,
= 0, else.

In this way we can again reduce a conjunction of two noun phrases to a conjunction of sentences.

7.2.2 A Problem for Collective Interpretation

But now consider the following example:

(21) Leopold and Stephen met.

If we apply the same technique to this case — that is, type raising of the names to (et)t and conjunction by *and* defined for this type — we get a statement that says that (21) is true if Leopold met and Stephen met, and false otherwise. But this is not what this sentence means. It is even hard to see what "Leopold met" should mean at all.

This problem was observed quite early. The Scottish philosopher James Beattie, in his work *The Theory of Language* (1783), phrases it as follows:

> So, when it is said, Peter and John went to the temple, it may seem, that the conjunction *and* connects only the two names, *Peter* and *John*; but it really connects two sentences, ¾ Peter went to the temple, ¾ John went to the temple. [But this is different from examples] like the following: *Saul and Paul* are the same, [...], There is war between England and France: Each of these, no doubt, is one sentence, and if we keep the same phraseology, incapable of being broken into two. For, if instead of the first we say, "Saul is the same ¾ Paul is the same", we utter nonsense; because the predicate *same*, though it agrees with the two subjects in their united state, will not agree with either when separate. [... And] if we say, "There is war between England ¾ there is war between France", we fall into nonsense as before; because the preposition *between*, having a necessary reference to more than one, cannot be used where one only is spoken of (p. 346-7).

Today we say that a sentence like *Leopold and Molly are asleep* contains a **distributive interpretation**; the predicate *be asleep* “distributes” over the two conjuncts, *Leopold* and *Molly.* And we say that *Leopold and Stephen met* contains a **collective predication**; the predicate *met* is true of the “collective” of Leopold and Stephen.

So far we have seen cases in which predicates were either interpreted as collective or as distributive. But there are cases in which we have an ambiguity between these interpretations:

(22)Leopold and Stephen lifted the piano.

This can mean that Leopold and Stephen together lifted the piano (collective interpretation), or that Leopold lifted the piano, and Stephen lifted the piano (distributive interpretation). With our current theoretical tools we can just handle the distributive interpretation. — Another example that is ambiguous is the following:

(23)Leopold and Molly received a letter.

Either there was one letter addressed to Leopold and Molly collectively, or there were two letters, one addressed to Leopold, the other to Molly. In this case the predicate *received a letter* applies to Leopold and Molly distributively.

7.2.3 Sum Individuals

The crucial property of the collective interpretation seems to be that we predicate VPs like *met* or *lifted the piano* to John and Mary together, or to three girls together. Formally, we should have entities in our model that correspond to the **collection** or the **sum individual** of John and Mary, or of three girls. So let us introduce sum individuals into the domain of discourse.

Let us assume that the domain of discourse D_e is structured in the following way: For any two entities x, y, there is a third entity that is the sum of x and y. We write x   y for that entity. Example: if LB and SD are entities in D_e, so is LB   SD. We may apply predicates to such entities:

(24) ⟦*Leopold and Stephen met*⟧ = ⟦*met*⟧(LB   SD)

What this says is that the sum individual consisting of LB and SD has the “met” property, which we can understand as saying that the parts of this sum individual met with each other.

What are the properties of the sum operation   ? First, notice that we can coordinate any two names. Hence   should map any two entities x, y in the domain D_e to an entity x   y. Furthermore, two entities x, y should be mapped to a unique sum individual. What this all means is that   is a **function** from $D_e \times D_e$ to D_e. Second, the order in which we combine two entities should not matter. *John and Mary met* is true exactly if *Mary and John met* is true. Functions with this property are called **commutative.**

(25)x   y = y   x, for all x, y D_e

In more complex sum formations the order in which we perform the sum operations should not matter. *John and* [*Mary and Sue*] *met* should be true if, and only if, [*John and Mary*] *and Sue met* is true. Functions with this property are called **associative**.

(26)x   [y   z] = [x   y]   z, for all x, y, z D_e

This allows us to omit brackets and write x   y   z instead. In natural language coordination, the constituents are typically treated as belonging to the same level, as in *John, Mary and Sue met.*

The sum of an entity with itself is that entity again. *John and John left* means the same as *John left*, if we are not talking about two different Johns here. Of course, a sentence like *John and John left* sounds quite strange, but this is simply because it means the same as the shorter sentence *John left*, and so it is unclear why the speaker did not choose this simpler sentence instead. Functions that have this property are called **idempotent**:

(27) x   x = x, for all x D_e.

The properties we have assumed for the sum operation   we have essentially stipulated. They seem to reflect the properties of *and*, as we need it for collective predications, and therefore they appear to be a good choice. Such basic assumptions of a theory (in this case, a theory of the sum operation) are called **axioms**.

With the help of   , we can define the relation of a **part**, in the sense that John is a part of John and Mary, or that Mary is a part of the students (if she is a student). Let us write ≤ for the **part relation**. We have the following definition:

(28) x ≤ y iff x   y = y, for all x, y D_e

The part relation has the following properties:

(29) a. For all x D_e, x ≤ x (**reflexivity**);
  b. For all x, y, z D_e, if x ≤ y and y ≤ z, then x ≤ z (**transitivity**);
  c. For all x, y D_e, if x ≤ y and y ≤ x, then x = y (**antisymmetry**).

These properties don't have to be stipulated; they follow from the way how we have defined the part relation (28) and the properties of the sum operation (25), (26) and (27). Take, for example, transitivity.

(30) a. To prove: if x ≤ y and y ≤ z, then x ≤ z.
  b. We assume x ≤ y and y ≤ z, and try to derive x ≤ z.
  c. x ≤ y means x   y = y, by definition.
   y ≤ z means y   z = z, by definition.
  d. As x   y = y, we can replace y in y   z = z, which gets us (x   y)   z = z.
  e. By associativity of   , we have (x   y)   z = x   (y   z) (= z).
  f. As y   z = z, we have x   (y   z) = x   z (= z).
  g. As x   z = z, we have x ≤ z, by definition.

Statements like (29.a,b,c) that can be derived from other, basic assumptions (the axioms) are called **theorems**.

Relations that are reflexive, transitive and antisymmetric are called **weak order relations**. Notice that the part relation includes identity, that is, every entity is a part of itself. We normally do not use the English word *part* in this way. But we can define a relation < that comes closer to the English use, the **proper part relation**:

(31) x < y iff x ≤ y and not y ≤ x, for all x, y D_e.

That is, x is a proper part of y iff x is a part of y, but y is not a part of x. By this definition the part relation has the following properties:

(32)a. For no x D_e: x < x (**irreflexivity**)
b. For all x, y, z D_e: If x < y and y < z, then x < z (**transitivity**)
c. For no x, y D_e: x < y and y < x (**asymmetry**).

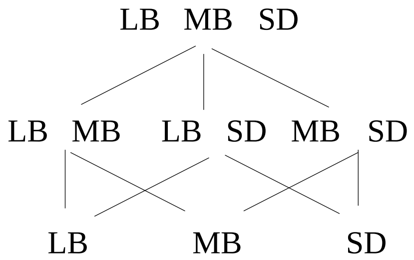
We want to be able to identify "simple" entities that don't have "proper" parts, such as LB. Call them **atoms**. I will use A for the set of atoms. It is defined as follows:

(33)A $=_{def}$ {x D_e | there is no y D_e such that y < x}

As an example of a universe with sum individuals, let us assume a universe D_e with three atoms LB, MB, SD. We then have that D_e has 7 elements all in all: D_e = {LB, MB , SD, LB MB, LB SD, MB SD, LB SD MB}. The set of atoms A is {LB, MB, SD}.

We can depict this structure in a diagram as below. We have used the following convention: If an entity x is part of another entity y, and there is no entity in between x and y, then x and y are connected, and x is below y. Such diagrams are called **Hasse diagrams**.

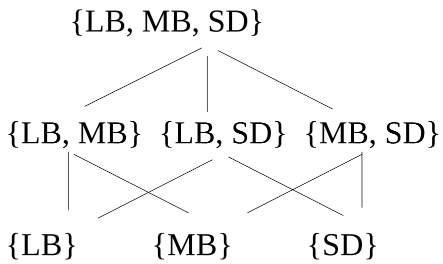
(34)



Structures like that are called **lattices**. More precisely, they are **join** lettices, as they only have the join operation . Regular lattices also have a so-called **meet** operation. The meet of two elements is what the two elements have in common. For example, the meet of LB SD and LB MB is LB. But not every two elements have a meet in our lattice; for example, LB and MB SD don't have a common part.

There is a way to model such a structure with set-theoretic means: If A is a set, then pow(A)-Ø is a suitable structure, where is rendered by , and ≤ is rendered by . Our example above then can be rendered in the following way:

(35)



In this model, singletons like {LB} represent the atoms. The join operation corresponds to set union, the part relation corresponds to the subset relation, and the meet operation corresponds to set intersection. As our representation does not contain the empty set, we do not have a general meet operation. Notice that the empty set would not correspond to any "real" object.

7.2.4 Treatment of Collective and Distributive Predication

Collective and distributive predication can be treated easily in this framework. The collective interpretation can be generated in the following way. First, we have to assume that *and* allows for the combination of two entities in D_e, which then is interpreted as the sum operation:

(36) If α, β D_e, then ⟦*and*⟧(α)(β) = α   β

Notice that this meaning of *and* is quite different from the generalized meaning defined in (17). The generalized meaning was defined on the basis of the meaning of *and* for sentences, type t, but (36), in a sense, specifies a new basic meaning. The meaning of *and* based on sentences is called **Boolean**, after George Boole, a British logician of the 19th century (and together with disjunction and negation we talk of "Boolean operators"). The meaning of *and* that is defined for entities of type e and is based on the sum operation is called **non-Boolean**.

We can now treat collective interpretations, as follows:

(37) a. ⟦[$_S$[$_{NP}$[$_{NP}$*Leopold*] *and* [$_{NP}$*Stephen*]] [$_{VP}$*met*]]⟧
b. = ⟦*met*⟧(⟦[$_{NP}$[$_{NP}$*Leopold*] *and* [$_{NP}$*Stephen*]]⟧)
c. = ⟦*met*⟧(⟦*and*⟧(⟦*Leopold*⟧)(⟦*Stephen*⟧)
d. = ⟦*met*⟧(⟦*Leopold*⟧   ⟦*Stephen*⟧)
e. = λx D_e[x met](LB   SD)
f. = [LB   SD met]
g. = 1, if LB   SD have the "met" property,
= 0, else.

Distributive interpretations can be derived as in (20) above, that is, by first type-lifting the names to quantifiers and then using Boolean *and* for quantifiers. I give another derivation here:

(38) a. ⟦[$_S$[$_{NP}$[$_{NP}$*Leopold*] *and* [$_{NP}$*Molly*]] [$_{VP}$ *are asleep*]]⟧
b. = ⟦[$_{NP}$[$_{NP}$ *Leopold*] *and* [$_{NP}$*Molly*]]⟧(⟦*are asleep*⟧)
c. = ⟦*and*⟧(⟦*Leopold*⟧)(⟦*Molly*⟧)(⟦*asleep*⟧)
d. = λQ $D_{(et)t}$λQ′ $D_{(et)t}$λP D_{et}[MIN{Q(P), Q′(P)}](⟦*Leopold*⟧)(⟦*Molly*⟧)(⟦*asleep*⟧)
e. = MIN{⟦*Leopold*⟧(⟦*are asleep*⟧), ⟦*Molly*⟧(⟦*asleep*⟧)}
f. = MIN{λP D_{et}[P(LB)](λx[x is asleep]), λP D_{et}[P(MB)](λx[x is asleep])}
g. = MIN{λx[x is asleep](LB), λx[x is asleep](MB)}
h. = MIN{[LB is asleep], [MB is asleep]}
i. = 1 if [LB is asleep] = [MB is asleep] = 1,
= 0, else.

But we could also think about another way of deriving the distributive interpretation, one that does not take the detour of type lifting of names to quantifiers and applying a type-lifted version of conjunction. We may assume the following: If a predicate like *be asleep* is applied to a sum individual x, then it gives us the truth value 1 only if *be asleep* applies to every atomic part of x. This appears to be quite natural; if a collection of persons is asleep this means that each of those persons is asleep. Let us call this the **distributive interpretation** of predicates like *be asleep.*

(39) Rule for ⟦*asleep*⟧:
For all x D_e, ⟦*asleep*⟧(x) iff for all y ≤ x: If y   A then ⟦*asleep*⟧(y).

Now we can have a slightly simpler derivation of the distributive interpretation, one that is as a matter of fact equivalent to the collective one:

(40) ⟦[$_S$[$_{NP}$[$_{NP}$*Leopold*] *and* [$_{NP}$*Molly*]] [$_{VP}$*are asleep*]]⟧
b. = ⟦*asleep*⟧(⟦[$_{NP}$[$_{NP}$*Leopold*] *and* [$_{NP}$*Molly*]]⟧)
c. = ⟦*asleep*⟧(⟦*and*⟧(⟦*Leopold*⟧)(⟦*Molly*⟧)
d. = ⟦*asleep*⟧(⟦*Leopold*⟧   ⟦*Molly*⟧)
e. = λx D_e[x is asleep](LB   MB)
f. = [LB   MB asleep]
g. = [LB is asleep] and [MB is asleep]
h. = 1, if [LB is asleep] = [MB is asleep] = 1,
= 0, else.

But what about cases in which we have a real ambiguity, as in *Leopold and Stephen lifted the piano*? We certainly cannot have a rule for *lifted the piano* that is similar to the rule for *asleep* in (39). But we may assume that there is an operator that enforces a distributive interpretation of the verb phrase. Evidence from that comes from the fact that English has an overt operator of this kind, *each*:

(41) a. Leopold and Stephen each lifted the piano.
b. Leopold and Stephen each got £100.

We can assume the following meaning for *each* and the silent distributive operator:

(42) ⟦*each*⟧ = λP D_{et}λx D_e[{y   A | y ≤ x}   P]

This means that the set of entities y that are atomic parts of x all have the property P. Consider the following derivation:

(43) a. ⟦[$_S$[$_{NP}$[$_{NP}$*Leopold*] *and* [$_{NP}$*Stephen*]] [$_{VP}$ *each* [$_{VP}$ *lifted the piano*]]]⟧
b. = ⟦*each*⟧(⟦*lifted the piano*⟧)(⟦*Leopold*⟧   ⟦*Stephen*⟧)
c. = λP D_{et}λx D_e[{y   A | y ≤ x}   P](⟦*lifted the piano*⟧)(LB   SD)
d. = {y | y ≤ LB   SD}   ⟦*lifted the piano*⟧
e. = 1 if ⟦*lifted the piano*⟧(LB) = ⟦*lifted the piano*⟧(SD) = 1,
= 0, else.

7.2.5 The Semantic Side of Number Agreement

We now have two ways of deriving the distributive interpretation: By type-lifting of names to quantifiers and applying Boolean conjunction, or by non-Boolean conjunction and a distributive operator. Is there any way to decide which of these options is preferred?

Actually, there is. We haven't paid attention so far to the fact that the verb shows number agreement with the subject. There is one basic difference between conjunctive NPs and disjunctive NPs in this respect: With the former, the verb typically shows plural agreement, and with the latter, singular agreement:

(44) a. Leopold and Molly are / ??is asleep.
b. Leopold or Molly ??are / is asleep.

This can be explained as follows: Disjunctive NPs like *Leopold or Molly* are always conjoined by a Boolean coordinator that reduces things to sentence disjunction, here *Leopold is asleep or Molly is* *asleep*. In those disjuncts the verb is singular, as the subject is singular. The singular in the original sentence presumably reflects the fact that the sentence is, in a sense, just a shorthand for a disjunction of two sentences with singular subjects.

Conjunctive NPs like *Leopold or Molly* can, in theory, be conjoined by non-Boolean *and* or by Boolean *and*. If they were conjoined by Boolean *and*, we would expect singular verb agreement, by the same reasoning. We don't, hence we should expect that the NPs are conjoined by non-Boolean *and*.

We can integrate this role of verb agreement into our representation as follows. First, subject-verb agreement is a feature of the auxiliary, or, more generally, the I^0 node (cf. chapter 5). We can intepret this feature as follows:

(45)a. $[\![SG]\!] = \lambda P \;\; D_{et} \lambda x \;\; A[P(x)]$
b. $[\![PL]\!] = \lambda P \;\; D_{et} \lambda x \;\; [D_e — A][P(x)]$

That is, the singular feature presupposes that the entity x is an atom, whereas the plural feature presupposes that x is not an atom (i.e., x is in the set D_e minus A). Now consider the following derivations. First, we have a case of non-Boolean conjunction.

(46)a. $[\![[_S [_{NP} [_{NP}$ *Leopold*$]$ *and* $[_{NP}$ *Molly*$]]$ $[_{I'}$ *are* $[_{VP}$ *asleep*$]]]]\!]$

b. = $[\![PL]\!]([\![$*asleep*$]\!])([\![[_{NP} [_{NP}$ *Leopold*$]$ *and* $[_{NP}$ *Molly*$]]]\!])$

c. = $\lambda P \;\; D_{et} \lambda x \;\; [D_e — A][P(x)]([\![$*asleep*$]\!])(LB \;\; MB)$

d. = $[\![$*asleep*$]\!](LB \;\; MB)$, provided that $LB \;\; MB \;\;\; D_e — A$, i.e. $LB \;\; MB$ is not an atom.

The condition is satisfied, and we get the same result as we had for (40) above. Now, consider an analysis of the same sentence under Boolean conjunction:

(47)a. $[\![[_S [_{NP} [_{NP}$ *Leopold*$]$ *and* $[_{NP}$ *Molly*$]]$ $[_{I'}$ *are* $[_{VP}$ *asleep*$]]]]\!]$
b. = $[\![[_{NP} [_{NP}$ *Leopold*$]$ *and* $[_{NP}$ *Molly*$]]]\!]([\![PL]\!]([\![$*asleep*$]\!]))$
c. = $\lambda Q \;\; D_{(et)t} \lambda Q' \;\; D_{(et)t} \lambda P \;\; D_{et}[\textsc{min}\{Q(P), Q'(P)\}]([\![$*Leopold*$]\!])([\![$*Molly*$]\!])([\![PL]\!]([\![$*asleep*$]\!]))$
e. = $\textsc{min}\{[\![$*Leopold*$]\!]([\![PL]\!]([\![$*asleep*$]\!])), [\![$*Molly*$]\!]([\![PL]\!]([\![$*asleep*$]\!]))\}$
f. = $\textsc{min}\{\lambda P[P(LB)](\lambda x \;\; D_e — A[[\![$*asleep*$]\!](x)]), \lambda P[P(MB)](\lambda x \;\; D_e — A[[\![$*asleep*$]\!](x)])\}$
g. = $\textsc{min}\{[\![$*asleep*$]\!](LB)], [\![$*asleep*$]\!](MB)]\}$, if $LB, MB \;\;\; D_e — A$

Now, the conditions that LB is not an atom and that MB is not an atom are not satisfied — they are atoms. But this means that we get neither 1 nor 0 as a result. The sentence violates the presupposition of the plural agreement. Of course, the presupposition of the singular agreement would be satisfied here, but singular agreement is only very marginally possible.

This can be seen as evidence that a sentence like *Leopold and Molly are asleep* is not formed via type-raising of the names to quantifiers and Boolean conjunction, but by non-Boolean conjunction. If this is so, we should assume that in cases in which non-Boolean conjunction is ruled out, and hence we have Boolean conjunction, singular number agreement becomes possible again. This is indeed the case:

(48)a. Every boy and every girl is / ?? are asleep.
b. No boy but every girl is / ?? are asleep.

Non-boolean conjunction is defined for entities of type D_e. Quantifiers like *every boy* or *no boy* are of type $D_{(et)t}$. Fore meanings of this type we only find Boolean conjunction. And interestingly we find singular agreement in these cases.

One might perhaps doubt that the type-raising from D_e to $D_{(et)t}$ is possible at all. However, notice that we can conjoin names and true quantifiers, which can be done only if we type-raise the name first. We have singular agreement in this case, just as predicted.

(49) Molly and every boy is asleep.

So it seems that type-lifting of names is possible after all. But then we should expect that Boolean conjunctions are possible, which means that the following example should be grammatical:

(50) ??Leopold and Molly is asleep.

It appears that the reason why (50) is not possible is that type-lifting of names is a “costly” operation, one that occurs only if there is no other way to derive a sentence, as in (49). If there is a simpler derivation, like using non-Boolean coordination, this is preferred, and this will block alternative, more costly derivations that would lead to (50).

7.3 Singular and Plural NPs

When we treated quantifiers like *three girls* or *most girls* in chapter 6 we did not really do anything about the fact that the nouns in these NPs are plural, in contrast to the nouns in *a girl* or *every girl*. We have seen that the number of verb phrases matters, and so we should expect that number in nouns plays a semantic role as well.

7.3.1 Singular and Plural Nouns

Let us start with number in nouns. A plausible assumption is that singular nouns apply only to atoms. We can say that *Mary is a girl* and that *Sue is a girl*, but we cannot say **Mary and Sue is a girl*. In a sense this is neither true nor false, which we take as indicating that the function that defines the meaning of *girl* is only defined for atoms. On the other hand, a plural noun should apply only to non-atoms. We can say *Mary and Sue are girls*, but not **Mary is/are girls*. This suggests meanings like the following:

(51) a. ⟦*girl*⟧ = λx A[x is a girl]

b. ⟦*girls*⟧ = λx D_e—A[x are girls]

We can derive the meaning of the plural *girls* from the meaning of the singular *girl* by the following definition:

(52) a. The set GIRLS is the smallest function that satisfies the following conditions:
(i) For all x D_e, if ⟦*girl*⟧(x) then GIRLS(x).
(ii) For all x, y D_e, if GIRLS(x) and GIRLS(y), then GIRLS(x   y).
b. ⟦*girls*⟧ = λx D_e—A[GIRLS(x)]

The first step (a) is a recursive definition of a function GIRLS. The first clause (i) states that everything that falls under *girl* also falls under GIRLS, and the second clause states that whenever there are two individuals x, y that fall under GIRLS, their sum falls under GIRLS as well. For example, if it is established that Jane and Sue both fall under *girl*, then it will follow that Jane   Sue falls under GIRLS. And if Mary falls under *girl* as well, it will follow that Jane Sue Mary fall under GIRLS. In mathematical parlance, GIRLS is the **closure** of ⟦*girl*⟧ under the sum operation . We also say that GIRLS is **cumulative** because whenever two entities fall under it, their sum falls under it as well.

Notice that even single girls will fall under GIRLS, due to clause (i). The second step (b) removes these cases by restricting the function to non-atoms.

Another way of defining the meaning of a plural noun from the meaning of its basic singular form is the following:

(53) ⟦*girls*⟧ = λx D_e—A[{y A | y < x} ⟦*girl*⟧]

That is, ⟦*girls*⟧ is a function from sum individuals x to truth values that maps x to 1 if all the atomic parts y of x fall under ⟦*girl*⟧, and else to 0.

It is actually debatable whether we should exclude atomic girls from the meaning of *girls*. Consider the following dialogue:

(54)a. Speaker A: Do you have children?

b. Speaker B: Yes, one. / *- No, one.

If the plural *children* does not apply to atomic children, then the answer *yes* or *yes, one* should be quite strange. Nevertheless, it is normal, and the denial *No*, or *No, (just) one* is decidedly peculiar. So we should assume that the plural *children* does include atomic children as well. But why, then, would it be strange for someone to say *I have children* if he or she in fact has just one child is because the sentence *I have one child* would be more informative in such a situation. Hence scalar implicature can account for that.

Another problem arises with nouns that denote **collectives**, like *couple*, *committee*, *rock band* or *battalion*. Take *committee*. A plausible assumption is to assume that it applies to people that form a committee. If Mary, Sue and Jane form the International Dinner Committee, we should be able to say:

(55)Mary, Sue and Jane are a committee.

But then we have a case in which a singular noun, *committee*, applies to a singular entity, contradicting our previous assumption.

What should we do? Fortunately, there is independent evidence that nouns like *committee* do not simply apply to the sum of the members of a committee. For example, the members of a committee can change; Jane might be replaced by John on the International Dinner Committee, and it stays the same committee. But the members of a sum individual cannot change; the sum individual Mary Sue Jane is distinct from Mary Sue John. Also, the same persons can form different committees; Mary, Sue and Jane could also be on the Defense Celebration Committee. But there is only one sum individual that represents the sum of Mary, Sue and Jane.

So it seems that committees, couples, battalions etc. are special kinds of entities that have members, but that are atoms, elements of A. For example, the International Dinner Committee is an atom in A that stands in a membership relation to Mary, to Sue, and to Jane. We can then distinguish between the committee itself and the sum of its members.

For our current purposes we should notice that it is not necessary to change the atomicity requirement for singular nouns like *committee*.

7.3.2 Existential Quantifiers

We can now treat sentences with plural NPs that have the determiner *some*, which can be used both for singular and plural nouns:

(56)a. $[\![[_S [_{NP} [_{Det}$ *some*$] [_N$ *girls*$]] [_{VP}$ *sang*$]]]\!]$
b. = ⟦*some*⟧(⟦*girls*⟧)(⟦*sang*⟧)
c. = $\lambda P \;\; D_{et} \lambda P' \;\; D_{et}[P \cap P' \neq \;\;]$(⟦*girls*⟧)(⟦*sang*⟧)
d. = [⟦*girls*⟧ ∩ ⟦*sang*⟧ ≠]
e. = $[\lambda x \;\; D_e$—A[{y A | y≤x} ⟦*girl*⟧] ∩ ⟦*sang*⟧ ≠]
f. = 1, if there is some x whose atomic parts consist of girls and which sang,
= 0, else.

This is the right interpretation; we get the value True if there is a sum individual consisting of girls that also has the property expressed by *sang*.

In many cases it is not even necessary to use an overt determiner, like *some*. The following two sentences have truth conditions that are pretty much identical:

(57)a. Some girls arrived.
b. Girls arrived.

This holds in particular if *some* is destressed (sometimes written *sm* in the linguistic literature). If *some* is stressed, it is often understood as referring to a subset of a group of girls (a reading that can be made more obvious by *some of the girls*).

To handle cases like (57.b) it is often assumed that we have an empty determiner which I will call *0* that is interpreted just like *some*. We then have the following interpretation:

(58)a. $[\![[_S [_{NP} [_{Det}$ *0*$] [_N$ *girls*$]] [_{VP}$ *arrived*$]]]\!]$
b. = ⟦*0*⟧(⟦*girls*⟧)(⟦*arrived*⟧)
c. = $\lambda P \;\; D_{et} \lambda P' \;\; D_{et}[P \cap Q \neq \;\;]$(⟦*girls*⟧)(⟦*arrived*⟧)
d. = [⟦*girls*⟧ ∩ ⟦*arrived*⟧ ≠]

We will see another potential use of the empty determiner *0* in the next section.

7.3.3 Number Words

In chapter 6 we have treated number words as determiners that form Generalized Quantifiers. For example, *three* is interpreted as follows:

(59) ⟦*three*⟧ = $\lambda P' \;\; D_{et} \lambda P \;\; D_{et}[\#(P' \cap P) \geq 3]$

That is, the number of elements in the intersection between the noun meaning P′ and the VP meaning P is at least three. We then have derivations like the following:

(60)a. $[\![[_S [_{NP} [_{Det}$ *three*$] [_N$ *girls*$]] [_{VP}$ *arrived*$]]]\!]$
b. = ⟦*three*⟧(⟦*girls*⟧)(⟦*arrived*⟧)
c. = [#(⟦*girls*⟧ ∩ ⟦*arrived*⟧) ≥ 3]

With the introduction of sum individuals we are facing a propblem here: What (60.c) literally says is that there are at least three entities that fall both under *girls* and under *arrived.* But not these entities could be sum individuals that could contain many girls. Obviously, this interpretation is on the wrong track.

One way to set it right is to disregard the number marking on *girls* and treat it as if it were *girl*, which applies only to atomic entities. We could say that number marking of nouns is a purely syntactic phenomenon, and that number words greater than one require a plural noun.[1] This is against our previous argument that number of nouns is semantically relevant.

Another way to go is to assume that number words aren't determiners, but adjectives, and that the role of the determiner is satisfied by the empty determiner *0* as in (58).

It is easy to give an adjectival meaning to number words in our framework with sum individuals. Let us assume a function AT that gives us the number of atoms that a sum individual consists of, its **atomic number**. We have, for example, AT(LB MD SD) = 3. This function can be defined as follows:

(61)For all x D_e, AT(x) = #{y A | y≤x}

That is, the atomic number of x is the cardinality of the set of atoms y that are a part of x.

Now we can interpret number words as expressions of type $D_{(et)et}$, just like adjectives:

(62) ⟦*three*⟧ = λP D_{et}λx D_e[MIN{P(x), AT(x) = 3}]

This allows derivations like the following:

(63)a. ⟦[$_S$ [$_{Det}$ *0*] [$_N$ [$_{AP}$ *three*] [$_N$*girls*]]] [$_{VP}$ *arrived*]]⟧
b. = ⟦*0*⟧(⟦*three*⟧(⟦*girls*⟧)(⟦*arrived*⟧)
c. = ⟦*0*⟧(λP D_{et}λx D_e[MIN{P(x), AT(x) = 3](⟦*girls*⟧))(⟦*arrived*⟧)
d. = ⟦*0*⟧(λx D_{et}[MIN{⟦*girls*⟧(x), AT(x)=3])(⟦*arrived*⟧)
e. = [λx D_{et}[MIN{⟦*girls*⟧(x), AT(x)=3] ∩ ⟦*arrived*⟧ ≠]
f. =1, if there is an x that falls under ⟦*girls*⟧, that has three atoms, and that falls under *arrived*;
= 0, otherwise.

That is, the sentence is true if there is sum individual consisting of three girls that arrived. Under the assumption that *arrived* is a distributive predicate, this has the same truth condition as the original formalization, using generalized quantifiers as in (60).

In particular, both representations state that a sentence like *Three girls arrived* need not be false if actually, four girls arrived. The GQ approach did this by explicitly stipulating that the number of girls that arrived is equal or GREATER than 3. In our new approach we need no stipulation like that. The formula in (63.f) gives us all we need. Notice that it just states that there be one individual consisting of three girls that arrived. There might be very well other, and more comprehensive groups of girls that arrived as well.

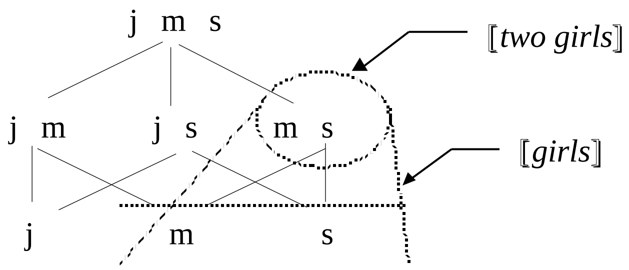
It is clear that predicates like *three girls* are not cumulative: If x is a sum individual of three girls, and y is a sum individual of three girls, then x y, in general, will not be a sum individual of three girls anymore. In addition, we find that whenever we have an individual x that falls under the

[1] Something like this has tobe done for decimal fractions, which always require the plural of the noun. Cf. e.g. *three point fifty dollars* or *zero point seventy-five dollars*, and even *one point zero dollars* (as contasted with *one dollar*).

predicate *three girls*, and we have a proper part y of x, then y will not fall under *three girls* anymore. In this property, *three girls* d⇔ers from *girls*. We say that *three girls* is **quantized**.

The meaning of the predicates *two girls* and *girls* can be illustrated in a Hasse diagram as follows (here I assume that m and s are girls):

(64)



7.3.4 The Definite Article

A strong argument in favour of the adjectival analysis of number words is the fact that there are NPs that contain a determiner and a number word:

(65)The three girls sang.

Obviously, the definite article *the* and the number word *three* cannot both fill the determiner role. In GQ theory it was sometimes assumed that *the three* forms a complex determiner (just like *both*, which has the same meaning as *the two*). But obviously a theory that gives a compositional interpretation to expressions like *the three girls* is to be preferred.

Our original approach to the meaning of the definite article was the following: It combines with a noun; it presupposes that the noun applies to exactly one entity, and it gives as a value this entity (cf. chapter 5).

(66) ⟦*the*⟧ = $\lambda P \quad D_{et}[\#(P) = 1 \mid \iota(P)]$

Now this is precisely the meaning that we would need for *the three girls*:

(67)a. ⟦[$_{NP}$ [$_{Det}$ *the*] [$_{N}$ [$_{Adj}$ *three*] [$_{N}$ *girls*]]]⟧
b. = ⟦*the*⟧(⟦*three*⟧(⟦*girls*⟧))
c. = ⟦*the*⟧($\lambda x \quad D_e$—A[MIN{⟦*girls*⟧(x), AT(x)=3}])
d. = ι($\lambda x \quad D_e$—A[MIN{⟦*girls*⟧(x), AT(x)=3}]),
provided that #($\lambda x \quad D_e$—A[MIN{⟦*girls*⟧(x), AT(x)=3}]) = 1, else undefined.

The condition here is that the function that maps entities x to 1 if x falls under *girls* and has the atomic number 3 can be applied to just one sum individual x. This is the case if there are exactly three girls in D_e. If there are fewer than three girls, then there is no such x, and the value of #(...) is 0. If there are more than three girls, there is more than just one such x, and the value of #(...) is greater than x. For example, if j, m, s and b are girls, then there are four individuals that satisfy the description *three girls*, namely j m s, j m b, m s b, and j s b.

But notice that we can use the definite article with a simple plural noun, too:

(68) *The girls sang.*

What does *the girls* refer to in this case? It refers to the sum individual that contains all the girls. But notice that the meaning of *the*, as defined in (66), does not give us that as a result.

(69)a. ⟦[$_{NP}$ [$_{Det}$ *the*] [$_N$ *girls*]]⟧
  b. = ⟦*the*⟧(⟦*girls*⟧)
  c. = ι(⟦*girls*⟧), provided that #(⟦*girls*⟧) = 1, else undefined.

Assume that we have three girls, j, m, and b. We would expect that the meaning of *the girls* is defined, and that it refers to j   m   b. But notice that the condition is not satisfied. If the noun *girls* applies to two or more girls, there are not one but four individuals that satisfy the description, namely, j   m, j   b, m   b, and j   m   b.

We must make it a part of the meaning of the definite article to pick out the **maximal** individual that falls under a description. Let us introduce a new notation of the sum operator   , as follows:

(70)If P D_{et}, then   P is the sum of all entities x such that P(x).

In a more formal definition, we can define   P as the **smallest** entity that contains **all** the entities in P as parts:

(71)   P = x iff
  a. for all y such that P(y) it holds that y ≤ x;
  b. for all z such that for all P(y) it holds that y ≤ z, it holds that x ≤ z.

If we render P as a set, we have, for example,   {j, m, b} = j   m   b. Notice that   P is undefined if P is empty.

Now we can give the meaning of the definite article as follows:

(72) ⟦*the*⟧ = λP D_{et}[P≠ | P]

That is, the definite article can be applied to predicates P of type D_{et}, provided that P is not empty, and gives us the sum of entities x that fall under P.

Let us derive the meaning of *the girls sang*:

(73)a. ⟦[$_S$ [$_{NP}$ [$_{Det}$ *the*] [$_N$ *girls*]] [$_{VP}$ *sang*]]⟧
  b. = ⟦*sang*⟧(⟦*the*⟧(⟦*girls*⟧))
  c. = ⟦*sang*⟧(λP D_{et}[P≠ | P](⟦*girls*⟧))
  d. = ⟦*sang*⟧((⟦*girls*⟧)), if ⟦*girls*⟧ ≠   , undefined else.
  e. = 1, if   (⟦*girls*⟧) sang,
    = 0, if   (⟦*girls*⟧) did not sing,
    undefined, if ⟦*girls*⟧ =

We have that ⟦*girls*⟧ =   if there is no girl, or if there is just one girl. In this case the sentence has no truth value; in all other cases, it is true if the sum of all girls sang (i.e., if every girl sang), and otherwise false.

However, the meaning of the definite article (72) doesn't give us the right result for cases like *the three girls sang*:

(74)a. $[\![[_S [_{NP} [_{Det}\ \textit{the}] [_N [_{AP}\ \textit{three}] [_N\ \textit{girls}]]] [_{VP}\ \textit{sang}]]]]\!]$
 b. = 1, if $([\![\textit{three}]\!]([\![\textit{girls}]\!]))$ sang,
 = 0, if $([\![\textit{three}]\!]([\![\textit{girls}]\!]))$ did not sing,
 undefined, if $[\![\textit{three}]\!]([\![\textit{girls}]\!]) =$

Where $[\![\textit{three}]\!]([\![\textit{girls}]\!])$ is a predicate that applies to sum individuals consisting of three girls. The interpretation here tells us that the sentence does not have a truth value if there aren’t three girls around, and this is fine. But intuitively we want it to be undefined if there are more than three girls around. However, this is compatible with the condition $[\![\textit{three}]\!]([\![\textit{girls}]\!]) =$: If there are four girls, then there are four distinct sum individuals that fall under *three girls*.

But notice that in this case the sum of all individuals that fall under *three girls* does not itself fall under *three girls*. While, on the other hand, the sum of all individuals that fall under *girls* certainly falls under *girls* again, due to the cumulativity of this meaning. This suggests the following meaning rule for the definite article:

(75) $[\![\textit{the}]\!] = \lambda P \quad D_{et}[P \neq \quad, P(\quad P) \mid \quad P]$

That is, it is required that the property P applies to the sum individual P. This will give us the following derivations for *the girls sang* and *the three girls sang*:

(76)a. $[\![[_S [_{NP} [_{Det}\ \textit{the}] [_N\ \textit{girls}]] [_{VP}\ \textit{sang}]]]]\!]$
 b. $= [\![\textit{sang}]\!]([[\![\textit{girls}]\!]([[\![\textit{girls}]\!] \neq \quad, [\![\textit{girls}]\!](\quad [\![\textit{girls}]\!]) \mid \quad ([\![\textit{girls}]\!])])$
 c. = 1, if $([\![\textit{girls}]\!])$ sang,
 = 0, if $([\![\textit{girls}]\!])$ did not sing,
 undefined, if $[\![\textit{girls}]\!] =$ or $[\![\textit{girls}]\!](\quad [\![\textit{girls}]\!]) = 0$

(77)a. $[\![[_S [_{NP} [_{Det}\ \textit{the}] [_N [_{AP}\ \textit{three}] [_N\ \textit{girls}]]] [_{VP}\ \textit{sang}]]]]\!]$
 b. = 1, if $([\![\textit{three}]\!]([\![\textit{girls}]\!]))$ sang,
 = 0, if $([\![\textit{three}]\!]([\![\textit{girls}]\!]))$ did not sing,
 undefined, if $[\![\textit{three}]\!]([\![\textit{girls}]\!]) =$ or $[\![\textit{three}]\!]([\![\textit{girls}]\!])(\quad [\![\textit{three}]\!]([\![\textit{girls}]\!]))$

We will also get the right result for *the girl sang*, if the meaning of *girl* applies to atomic girls only:

(78)a. $[\![[_S [_{NP} [_{Det}\ \textit{the}] [_N\ \textit{girl}]] [_{VP}\ \textit{sang}]]]]\!]$
 b. = 1, if $([\![\textit{girl}]\!])$ sang,
 = 0, if $([\![\textit{girl}]\!])$ did not sing,
 undefined, if $[\![\textit{girl}]\!] =$ or $[\![\textit{girl}]\!](\quad [\![\textit{girl}]\!]) = 0$

If there is exactly one girl x, everything is fine, as the sum of all entities that fall under *girl* is that girl x, and x is certainly a girl. If there are two girls x, y, then the sum of x and y does not fall under *girl* anymore (it falls under *girls*), and hence the meaning of the expression *the girl* is undefined.

7.4 Mass Nouns and Measure Phrases

7.4.1 Mass Nouns

Nouns in English, and in many other languages, come in two types: **count nouns** like *apple*, *girl*, *book*, and **mass nouns** like *water*, *sand*, *money*. They show important morphological and syntactic differences. For example, mass nouns have no plural form and cannot be combined with number words or the quantifier *every*.

In certain respects mass nouns like *water, sand, money* behave similar to plural nouns like *apples.* For example, they don't need an article or number word to form an NP (cf. *Apples were sold in the shop* / *Milk was sold in the shop*), and simple mass nouns are cumulative: if x and y fall under *water,* then x   y falls under *water* again.

We can assume that mass nouns refer to entities, just like count nouns. For example, *water* refers to quantities of water — that puddle here, the ocean over there, but also to parts of the puddle or parts of the ocean.

One crucial difference to plural nouns is that we do not have to assume that there are "minimal" parts. The mathematical structure for plural nouns can be derived by assuming that there are simple, atomic individuals, and that sum individuals consist of combinations of these atomic individuals. For mass nouns we cannot assume atomicity. Of course, we know that, for example, water consists of H_2O molecules, which could be considered the "atoms" of the predicate *water.* But that water consists of molecules (instead of a homogenous mass) is a relatively recent discovery, and it is certainly not dictated by the English language.

If there are no atoms, then the function AT that gives us the atomic number of an object is undefined. This explains several facts about mass nouns.

- First, the lack of atoms in the extension of mass nouns explains that they do not have a plural form (which contains in its definition the condition AT(x) ≥ 2). Their singular form obviously does not make use of AT either; the singular is simply the morphologically unmarked form.
- Second, the lack of atoms also explains why mass nouns cannot be combined with number words (as in **three water(s)*). Again, number words make use of the AT function, which is not defined for the entities that fall under a mass noun.
- Third, count nouns allow for distributive readings (cf. *The apples cost 50c each*), whereas mass nouns do not (cf. **The water costs 10c each*).

However, one should not take the lack of atoms in a "literal" sense. There are nouns that denote practically the same things, like *coins* and *change*, one of which is a count noun and the other one a mass noun. Also, there are mass nouns, like *furniture*, which quite clearly apply to things that have minimal parts (e.g., a minimal part of the furniture in this room is that chair over there, but not the front left leg of that chair). When we talk about atoms we rather refer to the way how we conceptualize things, and not how things "really" are.

Also, we often find that one noun can be used both as a count noun and as a mass noun. For example, *water* can be used as a count noun, as in *three waters*, especially in a restaurant context, where it refers to a quantity of water of customary size (e.g., a glass). And *lamb* can be used as a mass noun, referring to the meat or in general the substance that makes up a lamb or lambs, as in *We had lamb for dinner*).

7.4.2 Measure Phrases

Mass nouns cannot be combined with number words, but they can be combined with **measure phrases**, as in *three gallons of water* or *five ounces of gold*. Measure words like *gallon* or *ounce* relate to a way of singling out quantities of a certain size. They are based on so-called **additive measure functions** which have the following property:

(79) m is an additive measure function for the entities in a set S if the following holds:

a. m is a function from S to numbers;

b. there is a sum operation   for the elements of S such that the following holds:
For all x, y   S that don't have common parts (that is, there is no z with $z \leq x$ and $z \leq y$),
$m(x \;\; y) = m(x) + m(y)$

For example, *gallon* is an additive measure function in this respect. If x are 3 gallons of water and y are 5 gallons of water, and x and y are distinct quantities of water, then x   y are 8 gallons of water. Notice that the function AT is an additive measure function as well. Can you think of non-additive measure functions?

Assume that GALLON is the measure function for gallon. Then we can give the following interpretation to *three gallons of water*:

(80) a. ⟦*three gallons of water*⟧

b. = ⟦*three gallons*⟧(⟦*water*⟧)

c. = ⟦*gallons*⟧(⟦*three*⟧)(⟦*water*⟧)

d. = $\lambda n \;\; R^{+} \lambda P \;\; D_{et} \lambda x \;\; D_{e}$[MIN{GALLON(x) = 3, P(x)}](3)(⟦*water*⟧)

e. = $\lambda x \;\; D_{e}${MIN{GALLON(x) = 3, ⟦*water*⟧(x)}]

That is, we get a predicate that applies to entities x if they are water and measure three gallons.

We find measure constructions not only with conventionalized measures like *gallon.* One frequent type uses containers, as in *three glasses of water*. Another type makes use of some prominent feature of the entities to be counted, as in *fifty head of cattle*. Terms of this type are called **classifier constructions**, where *head* is called a **classifier**. In many languages, especially in East Asia, this is the predominant construction, that is, there are hardly any "count nouns" as in English.

7.5 Kind reference

NPs that consist of simple mass nouns or plural nouns (so-called **bare plurals** and **bare mass terms**) also can give rise to another reading. Consider the following difference:

(81) a. Dogs are faithful.

b. Dogs are sitting on my lawn.

The first sentence makes a claim about dogs in general -- the kind *Canis*. Such sentences are also called **generic**. The second sentence says something about particular dogs. We have similar reading distinctions with mass terms:

(82) a. Milk is healthy.

b. Milk was spilled all over the floor.

The difference in reading seems to be due to the nature of the verb phrase. The VP *be faithful* expresses something like a permanent property, whereas the the VP *be sitting on the lawn* expresses a temporary property, an event.

One rather influential theory for such sentences was proposed by G. Carlson (1977). Here are some of the ingrediences of this theory:

- We have to assume a special sort of entities, **kinds**. Bare plurals like *dogs* and mass terms like *milk* are names of kinds.

- Kinds are related to objects that realize them. We express this by a relation R; for example, if Fido is a dog, then we have R(f, d) (where f is the dog Fido, and d is the kind *Canis*.
- Predicates like *be faithful*, *be healthy* can be applied to objects as well as to kinds. If they are applied to kinds, then they express that the objects that realize the kind typically have the property in question. For example, *dogs are social creatures* is true because entities that realize the kind dogs are typically social creatures. But note that there are certain predicates that just apply to kinds, as e.g. *be extinct*. We cannot say, for example, **Fido is extinct*.
- Predicates like *be sitting on the lawn*, *be barking*, *be spilled all over the floor* do not apply to individuals directly, but only to temporal **stages** of individuals. For example, if Fido sat on the lawn between 3 pm and 4 pm, then the predicate *be sitting on the lawn* applies to the stage of Fido between 3 pm and 4 pm.
- Stages can be seen as a special sort of entity. Stages are related to both objects and kinds via a relation R*. For example, if s is the stage of Fido between 3 pm and 4 pm, then we have R*(s, f) and R*(s, d).
- Objects and kinds together are called **individuals**, and are contrasted with **stages**. Predicates like *be a social creature* are called **individual-level**, predicates like *be sitting on the lawn* are called **stage-level**.

Given all that machinery, Carlson can give the following analysis for sentences with different predicates and different noun phrases:

(83)a. Dogs are faithful.
⟦*faithful*⟧(d)
b. Fido is faithful.
⟦*faithful*⟧(f)
c. Dogs are sitting on the lawn.
$\lambda x \;\; D_e$[there is an s, R*(s, x), and ⟦*sit on the lawn*⟧(s)](d)
= [there is an s, R*(s, d), and ⟦*sit on the lawn*⟧(d)]
d. Fido is sitting on the lawn.
$\lambda x \;\; D_e$[there is an s, R*(s, x), and ⟦*sit on the lawn*⟧(s)](f)
= [there is an s, R*(s, f), and ⟦*sit on the lawn*⟧(f)]

Note that stage-level predicates, like *be sitting on the lawn*, are analyzed in such a way that they can be applied to individuals, but internally they reduce to predications about stages. Also, notice that they have an existential quantifier built into their meaning.

This analysis predicts a crucial difference between singular indefinite NPs, like *a dog*, and bare plurals. Consider the following example:

(84)A dog is sitting on the lawn, and a dog is not sitting on the lawn.

This sentence is not contradictory; it simply says:

(85)a. ⟦*A dog is sitting on the lawn and a dog is not sitting on the lawn*⟧
b. = ⟦*and*⟧(⟦*a dog is sitting on the lawn*⟧, ⟦*a dog*⟧(⟦*not sitting on the lawn*⟧))
c = MIN({⟦*a dog*⟧(⟦*sitting…*⟧}, ⟦*a dog*⟧(⟦*not*⟧(⟦*sitting on the lawn*⟧))})
d. = MIN({λP[⟦*dog*⟧∩P≠](λx[there is an s, R*(s,x) ⟦*sit…*⟧(s)]),
λP[⟦*dog*⟧∩P≠](λPλx[1-P(x)](λx[there is an s, R*(s,x) ⟦*sit…*⟧(s)]))})
e. = MIN({[there is an x and an s, R*(s,x) ⟦*sit…*⟧(s)]],
[there is an x and there is no s, R*(s,x) ⟦*sit…*⟧(s)]})

But contrast it with the following sentence:

(86)Dogs are sitting on the lawn, and dogs are not sitting on the lawn.

This sentence IS contradictory. This is brought out by our analysis.

(87)a. ⟦*Dogs are sitting on the lawn, and dogs are not sitting on the lawn.*⟧
b. = MIN({⟦*dogs are sitting …*⟧, ⟦*dogs are not sitting…*⟧})
c. = MIN({λx[there is an s, R*(s,x), ⟦*sit…*⟧(s)](⟦*dogs*⟧),
⟦*not*⟧(λx[there is an s, R*(s,x), ⟦*sit…*⟧(s)])(⟦*dogs*⟧)})
d. = MIN({λx[there is an s, R*(s,x), ⟦*sit…*⟧(s)](d),
λx[1-[there is an s, R*(s,x), ⟦*sit…*⟧(s)]](d)})
e. = MIN({[there is an s, R*(s,d), ⟦*sit…*⟧(s)], [there is no s, R*(s,d), ⟦*sit…*⟧(s)]})

Carlson also discusses cases where an inherently stage-level predicate is changed to an individual-level predicate (so-called **habituals**). For example, *bark* (or *be barking*) is inherently stage-level, as it is stages of dogs that bark. Example:

(88)a. Dogs are barking.
there is an s such that [R*(s, d) ⟦*bark*⟧(s)]

b. Fido is barking.
there is an s such that [R*(s, d) ⟦*bark*⟧(f)]

However, we can also express individual-level properties:

(89)a. Dogs bark.
b. Fido barks.

This means, for example, that dogs have the tendency to bark in certain situations. Carlson proposes that stage-level predicates can be shifted to individual-level predicates. But it is notoriously difficult to say anything about the truth conditions of these habitual individual-level predicates, as they typically allow for exceptions. For example, sentences like *Frenchmen eat horsemeat*, or *Texans carry guns,* will be considered true by many people, even though the exceptions clearly outnumber the non-exceptional cases.