1 Noun phrases as generalized quantifiers

In the first-order predicate language (FOL), quantifiers are analyzed as variable-binding operators that combine with a sentence and give a sentence. We could assign them the type $<t,t>$. 

1. a. Every student walks.  
   b. $(\forall x)[\text{student}(x) \rightarrow \text{walk}(x)]$  
   (quantifier $\forall x)_{<t,t>}$ [open sentence]$_t$

The semantic analysis of quantifiers has its roots in logic and mathematics. Predicate logic has two basic quantifying operators: $\forall x$ the universal quantifier and $\exists x$ the existential quantifier. However, the first order predicate logic has two major shortcomings, as far as the analysis of quantificational expressions in natural language is concerned (see Barwise and Cooper 1981):

(i) the syntactic structure of sentences in natural language differs from that of predicate calculus; in natural language, quantified nominal expressions form a syntactic unit, unlike in their logical translations;

(ii) natural language contains quantified expressions which cannot be rendered using the first-order quantifiers $\forall x$ and $\exists x$: e.g., *most.*

The solution, according to Barwise and Cooper (1981), is to define quantified nominal expressions as the combination of determiners plus set expressions (see also Cooper 1983, Keenan and Stavi 1986, Van Benthem 1983, among others) – as “generalized quantifiers”:

$\text{generalized quantifier} = \text{[DET} + \text{CNP]}$  

‘CNP’: common noun phrase

- This uniform analysis of NPs as generalized quantifiers is in the tradition of Montague’s PTQ (1973) (see also Montague 1970a), a typed higher-order logic that presupposes that a (determiner) quantifier like *every* and a common noun phrase like *student*, its restriction, form a single syntactic and semantic constituent, a generalized quantifier.

- The categorial type of the entire NP *every student* is $<e,t>,t>$: a function from a one-place predicate (represented as a property or a set) to a truth value.

- The term ‘determiner’ is used here in a loose, non-syntactic sense, and includes pre-determiners, post-determiners and numerals (see Barwise and Cooper 1981, p. 216, fn. 2).

- Semantically, DETs are functions from sets (CNPs) to sets of sets, the type of generalized quantifier: $<e,t>, <e,t>, t>$. 

Let us look at this proposal in more detail.

Problem 1: Structural differences between quantifiers in predicate logic and quantificational NPs in natural language

For the sentences in (a) below, predicate logic is consistent with our observations made so far about using function argument application as a semantic ‘glue’ in building up semantic values compositionally. We know what types to assign to the categories in the syntactic tree to match the structure we see in the predicate logic formula on the right. The predicate is interpreted as a function of type $e \rightarrow t$, and the term as its argument, of type $e$. So functional application gives us a truth value as the semantic value of the sentence.

2. a. John walks.  
   b. $\text{walk(john)}$

   S
   ┌─┐
   NP  VP
   └──┘
   John  walks

   S (formula) : t
   Predicate: $e \rightarrow t$  
   Term: $e$

   walk  john
b. Every student walks.

∀x [(student (x) → walk(x)]

For (b), the predicate logic translation has an entirely different structure than the syntactic phrase structure. The meaning of every student is not a constituent in the logical formula, although it is a constituent in the English sentence.

**every student:** Where in the formula below is the meaning of every student?

∀x [ student (x) → walk (x)]

It is all the parts with the exception of the predicate “walk”:

∀x [student (x) → walk (x)]

Problem: Predicate logic helps us express the truth-conditional content of quantified English sentences like *Every student walks*, but it does not capture the structure of such English sentences. The quantificational formulas of predicate logic do not fit into a compositional approach to natural language in which the meaning of a complex expression is built up as a function of the meaning of its parts and the way they are put together, following the compositionality principle. If we want the interpretation of sentences to be compositionally built up as a mirror image of the syntactic structures assigned to them, then quantificational formulas like ∀x [student (x) → walk (x)] do not allow this.

Since the principle of compositionality plays an important role in linguistic theory, it is desirable
(i) to develop a uniform treatment of quantified nominal expressions (like every student) and non-quantified nominal expressions (like John) and
(ii) to generalize over different kinds of quantified nominal expressions.

Problem 2: Natural-language quantifiers that cannot be expressed by first-order quantifiers

In first-order predicate language, we can express quantificational NPs in sentences like

(3) Every / some / a / no student walks. (numerical quantifiers)

but not

(4) Most / many / a lot / a few fish swim.

Take the following sentence:

Most fish swim.

It is not just difficult to represent the correct interpretation of a sentence such as this one in a first-order predicate language, it is literally impossible. Bach (1989:54) illustrates this point with the following example. We could extend first-order predicate logic with a new quantifier MOST and assign it the following truth conditions:

MOSTxPx is true iff for most individuals x in the universe of discourse U it is true that x has the property P.

We need to relate the two predicates fish(x) and swim(x) by means of some connective.

MOSTx [fish(x) ? swim(x)]

None of the first-order logic connectives will work. We could replace ? by ∧.

MOSTx [fish(x) ∧ swim(x)]

That would mean: most things are swimming fish.

Nor can we replace it by the material implication connective →, which is used to interpret every.

MOSTx [fish(x) → swim(x)]
That would mean: _most things_ are such that if they are fish they swim.

Let us construct the following model: Let us say that _most_ indicates something like “at least two-thirds”. Now, consider a model in which 100 fish live, of which 25 swim. Our intuition tells us that the example _Most fish swim_ should be considered false in such a world. On 25 assignments that assign x a thing that is a fish, [fish(x) \land swim(x)] will be true, on 75 other assignments to fish, that formula will be false.

All we need is to have enough other things that are not fish, to assign as denotations of x to outnumber the non-swimming 75 fish and the putative interpretation of the “formula” will be true. (Note that if g(x) is not a fish, by the laws of logic, the implication will still be true).

The problem lies in the fact that in first-order predicate language we always quantify over the whole domain of individuals. This does not work for all NPs in natural language. It does not work for _most_. The meaning of _most fish_ cannot be calculated from the meaning of ‘most individuals/things x’ (in the domain A). We do not directly quantify over individuals, but we quantify over the individuals that satisfy the property of being a fish:

The sentence _Most fish swim_ is true iff _[most of the individuals that are fish]_ swim.

The point here is that in order to judge the truth of _Most fish swim_, we must look not at the whole domain, but at just exactly the subpart of the domain that is comprised of fish in the domain. The sentence says something not about most things, but about most fish.

Facts like these give us an important clue about why natural languages have the syntactic category of common nouns like _fish_. Common nouns are expressions that give a natural basis for picking out those subsets of the domain that we want to quantify over in sentences like _Most fish swim._

Generally, as Barwise and Cooper (1981) put it, the determiner ‘lives on’ the denotation of the common noun. It is not possible to capture this dependency relation in first-order predicate language, because the interpretation for the quantifiers \( \forall x \) and \( \exists x \) in terms of the assignment function \( g \) is built on the idea of checking the application of the formula to all or _some_ individuals in the universe of discourse. However, the interpretation of a determiner quantifier like _most_ is dependent on the subset within the universe of discourse that is specified by the common noun with which it forms a _w_.

In assigning the interpretation to the sentence _Most fish swim_, we disregard all the individuals that are not in the set of fish. _Most_ involves a proportional meaning component, which is a reason why we cannot express the meaning of _most_ in first-order predicative language. If we tried to express its meaning in FOL, we would have to know the number of fish that exist in our model. We would have to count the extension of _fish_, e.g. \( \#(\text{fish}) \), and relate that number to the number of “positive” and “negative” instances where the predicate _swim_ obtains. E.g.,

\[
[ \text{Most fish swim} ] = \exists x_1, ..., x_n [\text{fish}(x_1) \land ... \land \text{fish}(x_n) \land \neg x_1 = x_2 \land ... \land \neg x_{n-1} = x_n \\
\text{(mutually distinct)} \land n > 1/2 N(\text{fish})],
\]

where \( [N]^{\text{M},g} (\text{fish}) = \# \{ d \in A \mid \text{fish}(d) = 1 \} \)

Note that “N(fish)” is a second order predication.

**Solution: NPs uniformly analyzed as generalized quantifiers**

We have seen that there is a difference between the syntactic phrase structure of quantified sentences and their translation in predicate logic. Is it possible to interpret such an independently motivated English syntactic structure compositionally?

It is, assuming we enrich our logical language with the notion of of Generalized Quantifier (GQ). One of the fundamental tenets of the Generalized Quantifier theory is that it “permit[s] logical syntax to correspond more closely to natural language syntax” (Barwise and Cooper 1981, p. 159; and also previous suggestions in Montague 1974).

The formal theory of generalized quantifiers was developed by mathematicians (see for instance Mostowski 1957, Lindström 1966). It is implicit in Montague (1970a, 1973), who introduced it into linguistics (see Partee 2005), and established in linguistic semantics by Barwise and Cooper (1981) as well as Keenan and Stavi (1986), as a framework for the investigation of the universal constraints on quantification.

A compositional interpretation of independently motivated syntactic phrase structures for quantified sentences in natural language was proposed in Montague’s logic (Montague 1970a), and best shown in his “Proper Treatment of Quantification” (PTQ) (Montague 1973), where a typed higher-order logic made it possible to interpret NPs like _the man, every man, a man, no man_ uniformly as semantic constituents—GENERALIZED QUANTIFIERS—an idea advocated by (Lewis 1970) at the same time (see Partee 2005).

Terminological note: In Montague’s analysis, T stands for Terms, which in the 1970s corresponded to NPs, today’s DPs or QPs.
• Syntactic nominal constituents (NP) corresponds to logical generalized quantifiers:

\[
\begin{align*}
\text{S: } & t \\
\text{NP: } & [<e,t>,t] \\
\text{VP: } & <e,t> \\
\text{DET: } & \text{generalized quantifier}
\end{align*}
\]

• DET \textit{every} as a function

\[
\begin{align*}
\text{NP: } & <[<e,t>,t]> \\
\text{DET: } & \text{CNP}<e,t> \\
\text{VP: } & <e,t>
\end{align*}
\]

How to think about it:

(i) What is the semantic type of the CNP argument \([\text{student}]\)?
(ii) What is the semantic type of the NP value \([\text{every student}]\)?
(iii) If the argument is of semantic type \(a\), and the value is type \(b\), then the function is of type \(a \rightarrow b\). So what is the type of the function \([\text{every}]\)?

When interpreted as a function, the meaning of a DET like \([\text{every}]\) applies to a CNP meaning like \([\text{student}]\) to give a generalized quantifier, the meaning of a NP like \([\text{every student}]\), which in turn is a function which applies to a VP meaning like \([\text{walk}]\) to give a sentence meaning (extension: truth value).

We have:

\[
\begin{align*}
\text{CNP: } & \text{type } e \rightarrow t \\
\text{VP: } & \text{type } e \rightarrow t \\
\text{NP: } & \text{type } [e \rightarrow t] \rightarrow t \\
\text{S: } & \text{type } t
\end{align*}
\]

Therefore:

\[
\begin{align*}
\text{DET: } & \text{type } [e \rightarrow t] \rightarrow [e \rightarrow t] \rightarrow t \\
\text{CNP} & \rightarrow \text{NP}
\end{align*}
\]

2 Determiners as functions and as relations between sets

(The core of the idea following Larson 1995, also de Swart 1998.)

\[
\begin{align*}
\text{relational (tripartite) view} & = D(A, B) \quad \text{every (orange, sweet)} \\
& (\text{Cooper 1987, van Benthem 1986, also May 1985}) \\
\text{functional view} & = [D(A)](B) \quad [\text{every (orange)}](sweet) \\
& (\text{Montague 1973})
\end{align*}
\]

It is simpler to think about DET meanings in relational terms, as a relation between a CNP-type meaning and a VP-type meaning, using the equivalence between a function that takes a pair of arguments and a function that takes two arguments, one at a time.

Step 1: Determiner meanings can be analyzed as relations between sets.

We treat determiners “as if” the compositional structure were a relation between a CNP-type meaning and a VP-type meaning:

\[
\begin{align*}
\text{S: } & t \\
\text{DET} & \text{CNP}<e,t> \\
\text{VP} & <e,t>
\end{align*}
\]

This view of generalized quantifiers can be found in Cooper (1987), van Benthem (1986), also May (1985), who take determiners like \textit{every, most, some} to denote binary relations between two sets of individuals (extensional point of view) (see Partee 1995:544). Or alternatively, a two-place relation between one-place, first-order predicates. The
determiner is a second order predicate, since it establishes a relation between two sets of individuals, the meaning of the common noun and the meaning of the VP.

On this relational view, every, as in Every student walks, denotes a binary (two-place) relation between two sets, the set of students [student] and the set of walking individuals [walk]. Every student walks is then true iff everything that is a student is included in the set of walking individuals:

\[
\text{[student]} \subseteq \text{[walk]}
\]

the set of students is a subset of the set of those who walk

By means of a Venn diagrams, this idea can be represented in the simplest way as follows:

Set-theoretic notation:

\[
\text{[ Every student walks ]} = \{ x \mid x \text{ is a student} \} \subseteq \{ x \mid x \text{ walks} \}
\]

In words: the set of entities that have the property of being a student is a subset of the set of entities that have the property of walking.

So on the relational view, determiners like every have the status of a two-place relation between two first-order predicates, which means that they are second-order binary relations (predicates).

Assuming that determiners are binary second-order relations, they have the following the general format:

\[
D_a(X, Y) \quad \text{or} \quad D_a(Y)(X)
\]

where \( A \) stands for the universe of discourse.

The semantic difference between different determiners lies in the specific relation between the two sets they denote. As a first approximation, every, for example, denotes a subset relation, some the relation of non-empty intersection and no the relation of empty intersection. It is common to write these meanings directly in the metalanguage, using some set-theoretical notation without an intermediary logical language:

\[
\text{every: } A \subseteq B \quad \text{(a relation between sets A and B, “Every A B”)}
\]

\[
\text{some: } A \cap B \neq \emptyset.
\]

\[
\text{no: } A \cap B = \emptyset.
\]

The generalized quantifier every man is the set of subsets \( X \) of the universe \( A \) such that the set \( [\text{man}] \) is subset of them:

\[
[\text{every man}] = \{ X \subseteq A \mid [\text{man}] \subseteq X \}
\]

\[
[\text{some man}] = \{ X \subseteq A \mid [\text{man}] \cap X \neq \emptyset \}
\]
Many of the quantifiers listed above (some, no) just impose a restriction for the interpretation \([\mathcal{N}] \cap X\), that is, the intersection of the noun meaning and the verb phrase meaning. For example, \(\text{at least two } N\) says that the number of elements in this set must be greater or equal than two. Such quantifiers are called cardinal (or weak) quantifiers, as it depends solely on the cardinality of the intersection whether they hold or not.

Singular definite description with \(\text{the}\):
\[
\text{the } N = \{X \subseteq A \mid X \cap [\mathcal{N}] = \emptyset\}
\]
\[
\text{the present king of France} = \{X \mid \text{card}(\text{present king of France}) = 1 \land \text{present king of France} \subseteq X\}
\]
\[
\text{the present king of France is bald} = \{X \mid \text{card}(\text{present king of France}) = 1 \land \text{present king of France} \subseteq \text{bald}\}
\]

‘There is exactly one king of France and every king of France is bald’.

A singular definite NP like \(\text{the king of France}\) comes with two conditions:

First, \(\text{there be a king of France}\) (existence condition).
Second, \(\text{there be not more than one king of France}\) (uniqueness condition).

Compare this with the standard (Russell’s) representation in predicate logic

The present king of France is bald.
\[
\exists x[\text{PKF}(x) \land \forall y(\text{PKF}(y) \rightarrow x = y) \land \text{bald}(x)]
\]

We might say that the sentence \(\text{The king of France is bald}\) comes with the requirement that there exists exactly one king, and as soon as this requirement is not satisfied—‘presupposition failure’—, the sentence in which \(\text{the king of France}\) occurs is not felicitous. Such requirements are called presuppositions. For definite NPs this suggests the following analysis:

\[\text{the } N: \quad \{X \subseteq A \mid \text{card}(\mathcal{N}) = 1 \land \mathcal{N} \subseteq X\}\]

In case of the ‘presupposition failure’, the interpretation function \([\mathcal{N}]\) is partial: some sentences have no truth value, and this raises the question whether and how a truth value is assigned to complex sentences in which a sentence without a truth value occurs.

(See also Russell (1950) vs. Strawson (1950) debate on this point: According to Russell’s analysis, this sentence is false, since it contains an existence claim to the effect that there is a present King of France. According to Strawson, an utterance of the sentence \(\text{The king of France is bald}\) in a world where there is no present King of France is neither true nor false; perhaps the sentence has a truth value gap, or perhaps it fails to express a
determinate proposition. In any case, Strawson held that this fact supported a referential interpretation of expressions like ‘The present King of France’. If there is no present King of France, in case of the ‘presupposition failure’, an utterance containing such an expression is somehow defective.)

The presuppositional analysis can be extended to the following quantificational expressions analyzed as generalized quantifiers:

- **both** \( N \) \( \{X \subseteq A \mid [N] \text{ is a subset of } X\} \), presupposition: \( \text{card}([N]) = 2 \)
- **neither** \( N \) \( \{X \subseteq A \mid [N] \cap X = \emptyset\} \), presupposition: \( \text{card}([N]) = 2 \)
- **the seven** \( N \) \( \{X \subseteq A \mid [N] \text{ is a subset of } X\} \), presupposition: \( \text{card}([N]) = 7 \)

The flat relational structure corresponds to the general tripartite structure for quantified sentences proposed by Lewis (1975), Kamp (1981), in Discourse Representation Theory (DRT), and also by Heim (1982) (see Bach et al. 1987, 1995 for a discussion).

### Example

**Every orange is sweet.**

- **Operator**: \( \text{DET} \)
- **Restriction**: \( (A, \text{ orange}(x)) \)
- **Nuclear Scope**: \( \text{sweet}(x) \)

The A argument of the determiner operator, the RESTRICTION, restricts the domain over which the B argument, NUCLEAR SCOPE, is evaluated. Tripartite structures involve RESTRICTIVE QUANTIFICATION. The restriction within the tripartite structure comes from the first argument of the quantifier, expressed by a CNP.

The tripartite structure serves as a unifying generalization over the common properties of a variety of (semantically distinct) but related quantificational structures (see also Bach, Kratzer and Partee 1987; Partee 1991a, 1995, p. 541). For example, it allows us to represent semantic similarities between determiner quantifiers like *every* and adverbial quantifiers like *always*.

### Example

**Oranges are always sweet.**

- **Operator**: \( \text{ADVERB} \)
- **Restriction**: \( (A, \text{ orange}(x)) \)
- **Nuclear Scope**: \( \text{sweet}(x) \)

The tripartite structure allows us to handle different types of quantification, including quantification with unselective binding, quantifying over several variables at once and quantifying over events or cases (Partee 1995, p. 584) (see below Lewis 1975).

### Step 2: Determiners as functions

In order to fit the binary branching structure, we can reanalyze relational tripartite structures just as we reanalyzed transitive verb relations. The determiner can be reanalyzed as denoting a function which applies to one set (the interpretation of the CNP) to give a function from sets to truth values, the characteristic function of a set of sets (the interpretation of the VP):

\[
\text{DET}: \quad \text{type } [e \to t] \to [[e \to t] \to t]
\]

Every: takes as argument a set \( A \) and gives as result \( [B \mid A \subseteq B] \):

- the set of all sets that contain \( A \) as a subset.
- Equivalently: \( [[\text{every}] (A) = [B \mid \forall x (x \in A \to x \in B)] \)

Some, \( a \): takes as argument a set \( A \) and gives as result \( [B \mid A \cap B = \emptyset] \).

No: (not first-order expressible): \( [A \cap B = A - B] \).

The functional perspective on generalized quantifiers presupposes restricted quantification. *Every orange is sweet* analyzed with *every* as a restricted quantifier:

\[
[\forall x: \text{orange}(x)] \text{ sweet}(x)
\]

It presupposes the existence of a syntactic and semantic constituent comprising the quantificational element and a restrictive argument.

Generalized quantifiers and tripartite structures are equivalent for most purposes: there is a straightforward relation between the analyses of determiners in terms of generalized quantifiers and tripartite structures, at least in the case of quantification over a single variable, and they both crucially involve restrictive quantification.

### 3 GQs, compositionality and uniformity

Montague’s analysis of NPs (they correspond to Montague’s \( T \) for Terms) depends on two principles: UNIFORMITY and COMPOSITIONALITY.

**Compositionality** implies that an NP has an independent meaning. NPs are independent syntactic units, and their meanings are the building blocks for the meanings of larger units. Compositionality leads to a semantic analysis of NPs as GQs, as sets of properties. Consider the following sentence and its translation in predicate logic:

\[ (1) \text{ Every man walks.} \]

\[ \forall x [\text{man}(x) \to \text{walk}(x)] \]

The meaning assigned to the expression *every man* in the above sentence is not independent. Compare
(2) Every man sleeps.
∀x[man(x) → sleep(x)]

The two sentences differ in walk and sleep and the constant factor of the meaning is:

(3) Every man does it.

The procedure is now to make a variable of the thing that is different and abstract over it, thus retaining the constant factor of the meaning. This results in the following (extensional) interpretation:

(4) \(\lambda P \forall x[man(x) → P(x)]\), where \(P\) is of type \(<e,t>\).

\(P\) is a second-order predicate variable over one-place predicates like walk or sleep. The above formula is a formula of a second-order language, which has not only individual variables, but also predicate variables and second-order quantifiers (that quantify over properties). The first-order predicate language contains individual variables, apart from individual constants (names) and predicate constants, and quantifiers that bind individual variables.

Lambda expressions allow us to analyze the semantic value of quantified NPs into a determiner meaning and a noun meaning.

English NPs               Lambda expressions
Every man                 translation into a typed language
Some man                  \(\lambda P \exists x[\text{man}(x) ∧ P(x)]\)
No man                    \(\lambda P \neg \exists x[\text{man}(x) ∧ P(x)]\)

These expressions are equivalent by lambda conversion to:

\(\lambda Q[\lambda P \forall x[Q(x) → P(x)]]\) (man)
\(\lambda Q[\lambda P \exists x[Q(x) ∧ P(x)]]\) (man)
\(\lambda Q[\lambda P \neg \exists x[Q(x) ∧ P(x)]]\) (man)

\([\text{every man }]\) = \(\lambda P \forall x[\text{man}(x) → P(x)]\)
\([\text{every man walks}]\) = \(\lambda P \forall x[\text{man}(x) → P(x)]\) (walk) ⇔ \(\lambda P \forall x[\text{man}(x) → \text{walk}(x)]\)

So we can define the semantic value of every, some/a and no as follows:

every \(\lambda Q[\lambda P \forall x[Q(x) → P(x)]]\)
some, a    \(\lambda Q[\lambda P \exists x[Q(x) ∧ P(x)]]\)
no        \(\lambda Q[\lambda P \neg \exists x[Q(x) ∧ P(x)]]\)

This double abstract gives determiners the type \(<e,t>, <e,t>, t>>\:

\(\lambda Q_{<e,t>}[\lambda P_{<e,t>} \forall x[Q_{<e,t>}(x) → \forall t_{<t,t,t>} P_{<t,t>} (x_\beta)]](\text{man}_{<e,t>})\) (walk_{<e,t>})

Another example:

Every violinist is happy (with GQ-type subject)
\([\text{every violinist}] = \lambda P \forall x[\text{violinist}(x) → P(x)]\) type: \(e → t\)
\([\text{is happy}] = [\text{happy}]\) type: \(e → t\)
\([\text{every violinist is happy}]\)
⇔ \(\lambda P \forall x[\text{violinist}(x) → P(x)] [\text{happy}]\)
⇔ \(\forall x[\text{violinist}(x) → [\text{happy}]\] type: \(t\)

In general, we may use set-theoretic notation combined with lambda-expressions for clarity and convenience. In terms of the set theoretic notation, with the argument \(A\) playing the role of the variable \(Q\) lambda-calculus, and \(B\) playing the role of the variable \(P\):

\([\text{every } ](A) = \{B \mid \forall x (x \in A \rightarrow x \in B)\}\)
\([\text{some } ](A) = \{B \mid \exists x (x \in A \land x \in B)\}\)
\([\text{no } ](A) = \{B \mid \exists x (x \in A \land x \in B)\}\).

\([\text{every } ]\) takes as argument a set \(A\) and gives as result \([B \mid A \subseteq B]\): the set of all sets that contain \(A\) as a subset.
\([\text{some } ]\) takes as argument a set \(A\) and gives as result \([B \mid A \cap B \neq \emptyset]\).
\([\text{no } ]\) takes as argument a set \(A\) and gives as result \([B \mid A \cap B = \emptyset]\).

The effect of UNIFORMITY is twofold. First, it is standard to regard expressions as belonging to the same syntactic category when they exhibit the same (or similar) syntactic behavior, i.e., obeying the same distributional laws, whenever this can be syntactically determined. For this reason, both proper names (John) and definite descriptions (the tallest man in the world), on the one hand, and quantified NPs, on the other hand, are classified as NPs, though their semantic behavior is different.

Second, a syntactic category corresponds to one semantic type, that is, all expressions of a given syntactic category have the same kind of meaning. In the case of NPs, this meant that the useful analysis of quantified NPs as sets of properties was extended to proper names.
4 Proper names as generalized quantifiers

Quantified NPs are non-referential in the sense that they do not denote individuals, so they cannot be of type e. They denote sets of sets of individuals, or sets of properties, which means that their basic type is: \( <<e,t>,t> \).

In a type-logical approach, proper names may be assigned the type e of individuals: they refer to entities in our universe of discourse, but they can also be treated as generalized quantifiers. The relation between individual denoting NPs and GQs can be established by means of a rule of a type lifting:

\[ e \rightarrow <<e,t>,t> \text{ type lifting (Partee 1986)} \]

For example, \( \text{John} \) is interpreted as the set of sets to which John belongs or the set of properties John has.

\[
\llbracket \text{John} \rrbracket = \lambda P[j] = \{ X \subseteq A \mid \text{John} \in X \}
\]

The lambda abstraction is over a variable that is not of type e, but \( <<e,t>,t> \):

\[
\llbracket \text{John} \rrbracket = \lambda P_{<<e,t>,t>}[P_{<<e,t>,t>}](j_e)]
\]

In this way all NPs, that is both quantified and non-quantified NPs, are assigned the same semantic type, with which we associate a full second order interpretation.

\[
\text{John} \quad \lambda P[P(j)] \\
\text{John walks} \quad \lambda P[P(j)](\text{walk}) \iff \text{walk (j)}
\]

\begin{align*}
\text{every student} & \quad \lambda P[\forall x (\text{student}(x) \rightarrow P(x))] \\
\text{every student walks} & \quad \lambda P[\forall x (\text{student}(x) \rightarrow P(x))](\text{walk}) \\
& \quad \equiv \forall x (\text{student}(x) \rightarrow \text{walk}(x)) \\
\text{a student} & \quad \lambda P[\exists x (\text{student}(x) \land P(x))] \\
\text{the king} & \quad \lambda P[\exists x (\text{king}(x) \land \forall y (\text{king}(y) \rightarrow y = x) \land P(x))] \\
& \quad \text{(the set of properties which the one and only king has)}
\end{align*}

5 Adverbal Quantification

5.1 Adverbial quantification over events

So far we have paid attention to cases in which quantification is triggered by quantified NPs (or DPs, QPs in today’s terminology). There are many cases in which quantification is triggered by a quantificational adverbial (or adverb of quantification).

(1) When Mary recites a poem, she always/usually/often/sometimes/rarely/never speaks slowly.

We may analyze this sentence as involving quantification over events or situations. So a sentence like (2a) may be paraphrased as (2b):

\begin{align*}
\text{(2a)} & \quad \text{John buttered the toast in the bathroom with a knife at midnight} \\
\text{(2b)} & \quad \text{John buttered the toast in the bathroom with a knife at midnight, as in (1)}. \text{ This idea was introduced by Davidson (1967).}
\end{align*}

In order to express this paraphrase formally, we have to assume that verbs like butter or speak are related to an event (in addition to their relation to the ‘recitor’ and the recited object, like a poem). This idea was introduced by Davidson (1967). The Davidsonian theory is a cluster of theories about predicates/relations, their arguments, and modifiers. Three novel ideas:

\begin{enumerate}
\item The predicate butter (as in John buttered the toast in the bathroom with a knife at midnight) is a 3-place relation with an event argument (instead of being treated as a 2-place relation);
\item the event argument is existentially quantified; and
\item the modifiers are predicates of the event argument, added conjunctively.
\end{enumerate}

According to Davidson, verbs of action such as buttered involve implicit existential quantification over events. So n-place action verbs of tensed sentences are represented by (n+1)-place predicates, where the extra variable is a variable ranging over actions (a type of event). Action sentences are represented with explicit (first-order) existential quantification over an event argument, implying that they are indefinite descriptions of events.

\begin{align*}
\text{(3)} & \quad \text{Jones buttered the toast.}
\end{align*}
First-order predicate language: \texttt{butter (jones, the_toast)}  
\texttt{butter: 2-place predicate}

Davidson (1967): \texttt{(3x) (butter (jones, the_toast, x)}  
\texttt{[ see example (17) in Davidson 1967]}  
\texttt{butter: 3-place predicate}

Current notation: \texttt{∃e [butter (jones, the_toast, e)]}  
\texttt{‘e’: the event argument}  
\texttt{In words: There exists some event e which was a buttering of the toast by Jones (ignoring tense).}

The use of existential quantification implies that ordinary action sentences presuppose an ontology of events through the use of verbs: When we use action verbs, we implicitly refer to events.

The ontological status of events is controversial, but in any case they are somehow related to temporal and spatial locations, and that they typically have participants (e.g. an event of hitting has a hitter and a hitten object as participants). We will simply assume that events are entities of a specific sort in the domain and use the variable letters \(e, e'\) etc. for them.

Going back to our initial example (1): A verb like \textit{recite} is represented by a three-place predicate that relates an event, a person, and an object (e.g. a poem) to one another, and a verb like \textit{speak} is represented by a two-place predicate that relates events to persons, written as:

\begin{align*}
\text{(4) a. } [\text{ recite }] & = \lambda e \lambda x \lambda y \ [\text{recite}(y,x,e)] \text{ or} \nonumber \\
\text{b. } [\text{ recite }] & = \lambda e \lambda x \lambda y \ [\text{recite}(e,x,y)]
\end{align*}

A sentence then expresses the existence of an event:

(5) Mary recited a poem.  
\(\exists e, y [\text{recite}(e,m,y) \land \text{poem}(y)]\)

We can also represent the past tense of our example with the help of a relation \texttt{BEFORE(e,NOW)} and a time \texttt{NOW}, which refers to the time of utterance, by adding a clause \texttt{BEFORE(e,NOW)} in the scope of the existential quantifiers.

(6) Mary recited a poem.  
\(\exists e, y [\text{recite}(e,m,y) \land \text{poem}(y) \land \text{BEFORE}(e,\text{NOW})]\)

Davidson (1967) provided two main arguments for the analysis of action sentences with the event variable \(e\). First, the semantics of certain adverbials, like temporal and spatial adverbials, can be treated as predicates over events,

(7) Mary recited a poem on the porch at midnight.  
\(\exists e, y [\text{recite}(e,m,y) \land \text{poem}(y) \land \text{on-the-porch}(e) \land \text{at-midnight}(e)]\)

which allows us to represent the entailment relationships among sentences:

(8) Mary recited a poem on the porch at midnight.  
\(\Rightarrow \) Mary recited a poem on the porch.  
\(\Rightarrow \) Mary recited a poem at midnight.  
\(\Rightarrow \) Mary recited a poem.

This is because the following entailment relationships hold in predicate logic:

(9) \(\exists e, y [\text{recite}(e,m,y) \land \text{poem}(y) \land \text{on-the-porch}(e) \land \text{at-midnight}(e)]\)  
\(\Rightarrow \exists e, y [\text{recite}(e,m,y) \land \text{poem}(y) \land \text{on-the-porch}(e)]\)  
\(\Rightarrow \exists e, y [\text{recite}(e,m,y) \land \text{poem}(y) \land \text{at-midnight}(e)]\)  
\(\Rightarrow \exists e, y [\text{recite}(e,m,y) \land \text{poem}(y)]\)

Second, we can refer to events using demonstratives and definite NPs:

(10) a. Mary recited a poem. \(\text{It happened at midnight.}\)  
\text{b. The reciting was impressive.}

Such theoretical and empirical assumptions then motivate the following quantificational representation of (2a) (we represent \textit{speak slowly} as one, simple predicate \textit{\text{speak-slowly}}).

(11) When Mary recites a poem, she always speaks slowly. \([=(2a)\] ALWAYS\)  
\(\forall e [\exists y [\text{poem}(y) \land \text{recite}(e,m,y)] \rightarrow \text{\text{speak-slowly}(e,m)\]}

For other quantificational adverbials, quantificational adverbials like \textit{usually} or \textit{often}, we can make use of the power of generalized quantifiers. Let us introduce the following generalized quantifiers in our logical language (GQs viewed in relational terms, represented by means of tripartite structure):

\begin{align*}
\text{[always]} & = [\text{EVERY}(P)(Q)] = 1 \text{ iff } \lbrack P \rbrack \text{ is a subset of } \lbrack Q \rbrack \\
\text{[usually]} & = [\text{MOST}(P)(Q)] = 1 \text{ iff } \#(\lbrack P \rbrack \cap \lbrack Q \rbrack) > 1/2 \#(\lbrack P \rbrack) \\
\text{[often]} & = [\text{MANY}(P)(Q)] = 1 \text{ iff } \#(\lbrack P \rbrack \cap \lbrack Q \rbrack) > r \#(\lbrack P \rbrack), \text{ where } r \text{ is some contextually given proportion} \\
\text{[sometimes]} & = [\text{SOME}(P)(Q)] = 1 \text{ iff } \lbrack P \rbrack \cap \lbrack Q \rbrack \neq \emptyset \\
\text{[never]} & = [\text{NO}(P)(Q)] = 1 \text{ iff } \lbrack P \rbrack \cap \lbrack Q \rbrack = \emptyset \\
\end{align*}
Assuming a GQ theory, we can represent our example (1) as follows:

(12) When Mary recites a poem, she always speaks slowly.

\[ \text{EVERY}(\lambda x [\exists y \text{poem}(y) \land \text{recite}(e,m,y)](\lambda x [\text{slowly}(e,m,x)])] \]

5.2 Adverbial quantification over objects

Quantification over events may not always be sufficient to analyze certain cases of adverbial quantification. For example:

(1) A student of linguistics often knows foreign languages.

Following Davidson’s (1967) original proposal, stative verbs like know are often taken not to be related to an event (see e.g., Kratzer 1995). Only eventive (or episodic or dynamic) verbs like recite are. The terminology INDIVIDUAL-LEVEL PREDICATE vs. STAGE-LEVEL PREDICATE, introduced by Carlson (1977), is also used for the distinction between predicates that fail to introduce an event variable into the logical representation and those that do.

Assuming that stative verbs or INDIVIDUAL-LEVEL PREDICATES like know do not introduce an event variable, then there is no event that is related to a person knowing foreign languages in (1). It seems that in these cases, where we do not have events to quantify over, we quantify over variables that come with the indefinite NP, here over objects (i.e., here students):

(2) \[ \text{MANY}(\lambda x [\text{student-of-linguistics}(x)](\lambda x [\text{know-foreign-languages}(x)])] \]

Adverbial quantification can concern more than one individual, as in the following example:

(3) If a farmer has a donkey, he usually likes to beat it.

The most obvious reading can be paraphrased as follows, which involves quantification over pairs of variables x and y, or ‘cases’:

(4) adverbs of quantification as quantifiers over “cases”:

For most cases of x = farmer and y = donkey, if x has y, then x likes to beat y.

The quantification ‘over cases’ was introduced by Lewis (1975).

(5) Tripartite structure in Lewis (1975, p. 9):

a. A man who owns a donkey always beats it.
   b. Always [if x is a man, if y is a donkey, and if x owns y at time t], x beats y.
   c. x beats y

(5a) has a TRIPARTITE STRUCTURE consisting of the universal quantifier always, a restriction expressed by the three if-clauses, and a nuclear scope x beats y.

The restriction specifies the cases to quantify over, namely the man-donkey-time triples. So always quantifies over the admissible “cases” that satisfy the three if-clauses: namely, a value for x, a value for y, and a time-coordinate t. The “cases” are here the triples of a man, a donkey, and a time such that the man owns the donkey at some contextually specified time.

(5a) is true iff the sentence (5c) x beats y is true in all such admissible cases, or iff all the triples that satisfy the restriction also satisfy the nuclear scope. In the logical representation, this is reflected by the fact that the universal quantifier always binds all the variables x, y and t, producing universal quantification over man-donkey-time cases. Most importantly, this means that the universal quantifier always here unselectively binds all the variables x, y and t in its scope:

(6) Lewis’s (1975, p. 7) definition of unselective universal quantification:

\[ \forall \Phi \text{true iff } \Phi \text{ is true under every admissible assignment of values to all variables free in } \Phi. \]

“[I]t may well be a universal that any language that has conditional constructions will use them as one means of expressing quantification (a generalization that in fact follows if one accepts David Lewis’s suggestion that the basic function of if-clauses is to restrict operators, a suggestion which Kratzer has argued strongly for and exploited in a number of works, including Kratzer 1986, 1989)” (Partee 1990:12).

Consequence: Among the quantificational devices of English, we find determiners like every and also adverbs like always; modals like must are also assimilated to quantifiers (Partee 1995:543).

In Lewis’s example above, we have seen that always binds variables of various sorts, individual and temporal, and it unselectively binds all the free variables in its scope. Once we admit quantification over “cases” in our semantic representation language, w may represent the meaning of our ‘donkey’-sentence as follows:

(7) a. If a farmer has a donkey, he usually likes to beat it.
   b. MOST(\lambda x, y>[\text{boy}(x) \land \text{toy}(y) \land \text{has}(x,y)](\lambda x, y>[\text{like-to-beat}(x,y)])

Generally, an expression like
\[ \lambda <x,y>[\phi(x,y)] \]

where \( \phi(x,y) \) is a sentence, is interpreted as the set of pairs \( <x,y> \) that satisfy \( \phi(x,y) \) (we could also write \( [<x,y> \cup \phi(x,y)] \), “most pairs of a farmer \( x \) and a donkey \( y \)...”) 

However, our example might have another reading. It might be that what we want to say is:

(8) Most farmers who have a donkey like to beat it.

And there are subtle meaning differences between

(9) a. For most cases of \( x = \) farmer and \( y = \) donkey, if \( x \) has \( y \), then \( x \) likes to beat \( y \)

b. Most farmers who have a donkey like to beat it.

Imagine a situation in which there are ten farmers, one of them has 100 donkeys and likes to beat all of them, and the nine others have one donkey and do not like to beat it.

With respect to this situation, we have

reading 1: 109 farmer-donkey pairs, 100 positive cases, hence sentence is true.
reading 2: 10 farmers, only 1 positive case, hence sentence is false.

An independent question is how the reading 2 should be expressed in our semantic representation language. When we try to do it straightforwardly, we run into a problem, as the anaphoric binding between \( \text{donkey} \) and \( \text{it} \) cannot be expressed. Note that the indicated occurrence of \( y \) is unbound in the following formula:

(10) \text{MOST}(\lambda x \exists y [\text{farmer}(x) \land \text{donkey}(y) \land \text{have}(x,y)])(\lambda x [\text{like-to-beat}(x,y)])

There are various proposals to overcome this problem, following some original suggestions in particular Discourse Representation Theory (invented by Hans Kamp 1981), File Change Semantics (Irene Heim, 1982) and in general theories of “dynamic semantics” (see e.g., Chierchia 1995).

References


Partee, Barbara H. 1996. The development of formal semantics in linguistic theory. In The