In Relational Methods in Computer Science, hrsg. von Chris Brink, Wolfram Kahl und Gunther Schmidt, Wien: Springer, 229 - 249.

# Chapter 15

# Natural Language

*Michael Böttner*[1]

The purpose of this chapter is to show how relational algebra can be applied to the semantics of natural language. The use of relational algebra for natural language semantics was first proposed in [Suppes 1976] under the heading of *relational grammar* in the context of model theory. It was extended to various areas of English such as modification of nouns by adjectives [Suppes, Macken 1978], intonation [Suppes 1979b], rules of natural language inference [Suppes 1979a], anaphoric pronouns [Böttner 1992b] and [Böttner 1996], and coordination [Böttner 1994]. An extension to a procedural semantics was proposed in [Böttner 1992a]. Most of Suppes' articles have now become easily accessible in [Suppes 1991].

The organization of this chapter is as follows. In Sect. 15.1 the use of algebraic methods in logic and linguistics is outlined. In sections 15.2 and 15.3 an introduction is given to relational grammar in the context of model theory, and in Sect. 15.4 the notion of relational grammar is extended to procedural semantics.

## 15.1 Algebraic semantics for natural language

From the beginning of the development of algebra in mathematics there have been attempts to apply algebra to the analysis of natural language. One goal of these attempts was to be able to calculate with natural language expressions very much in the same way as one could already do with numbers in arithmetic, or to put it in modern terms, to do automated theorem-proving. Another goal was to reduce the grammar of a language to a "rational" grammar. Both projects have usually been attributed to Leibniz under the names of *characteristica universalis* and *calculus ratiocinator*. Characteristica universalis is a language regimented to allow to express every possible thought or concept. Calculus ratiocinator is a system of operations that allows to check arguments and logical inferences by computing. The main ideas for these projects had already been present in his dissertation published in Leipzig 1666 but can also be found in many of his later works. For a good introduction to both projects and further reference see [Parkinson 1966], [Burkhardt 1980], or [Ishiguro 1990]. But it was not until two centuries later that progress was made in the area of algebraic logic by Boole, De Morgan, Peirce, and Schröder. Traditional logic was intimately connected with the structure of

[1]Supported by Max Planck Institute for Psycholinguistics, Nijmegen.

natural language and this tradition was also alive in 19th century logic. With the rise of quantifier logic, however, from the *Principia Mathematica* of Russell and Whitehead onward ([Whitehead, Russell 1910]), logic became more and more removed from natural language. But in somewhat the same way as algebraic logic again came to the fore in the second half of this century, an algebraic approach to language has also been re-established.

The operations of Boolean algebra have been illustrated by natural language examples from its very beginnings. An example used by Boole (1854) is

$$z(x + y), \tag{15.1}$$

where $z$ stands for *European*, $x$ stands for *men*, $y$ stands for *women*, $+$ stands for *and*, and juxtaposition of terms stands for modifying a noun by an attributive adjective. With this interpretation (15.1), translates into

$$\textit{European men and women.} \tag{15.2}$$

Applying the law of distributivity to (15.1) yields

$$zx + zy, \tag{15.3}$$

and with the interpretation chosen for (15.1), (15.3) translates into

$$\textit{European men and European women.} \tag{15.4}$$

The algebraic equality between (15.1) and (15.3) then guarantees the semantic equivalence of the expressions (15.2) and (15.4). Algebraic symbolism like (15.1), (15.3) may thus be used as an instrument to compute inferences and equivalences of natural language expressions. Not just phrases like (15.2) but whole sentences can be treated in Boolean terms. Standard examples are sentences occurring in Aristotelian syllogisms. A *syllogism* is an argument with two premises and a conclusion, each of which is a sentence that has one of the following four forms:

$$\begin{array}{ll} S\ \mathbf{a}\ P\text{:} & \textit{All } S \textit{ are } P \\ S\ \mathbf{i}\ P\text{:} & \textit{Some } S \textit{ are } P \\ S\ \mathbf{e}\ P\text{:} & \textit{No } S \textit{ are } P \\ S\ \mathbf{o}\ P\text{:} & \textit{Some } S \textit{ are not } P \end{array} \tag{15.5}$$

A sentence exhibiting one of these forms is called *categorical* and is usually identified by a letter indicating whether the structure is affirmative universal (**a**), affirmative particular (**i**), negative universal (**e**), or negative particular (**o**). A truth condition in terms of Boolean operations can be spelt out for each of these structures as follows:

$$\begin{array}{lll} S\ \mathbf{a}\ P & \text{is true iff} & S \subseteq P \\ S\ \mathbf{i}\ P & \text{is true iff} & S \cap P \neq \emptyset \\ S\ \mathbf{e}\ P & \text{is true iff} & S \subseteq \overline{P} \\ S\ \mathbf{o}\ P & \text{is true iff} & S \cap \overline{P} \neq \emptyset. \end{array} \tag{15.6}$$

The semantics of a language is studied by model theory. Originally, model theory was applied to languages of logic, but once rigorous methods had become available for the syntax of natural language, model theory was successfully applied in this domain as well. A major breakthrough in natural language semantics was the work by

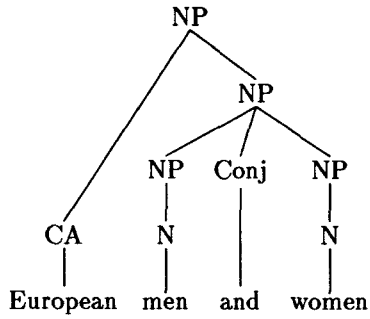



Fig. 15.1 Derivation tree for *European men and women*

Richard Montague [Montague 1974]. He proposed to view any natural language as a pair of two homomorphic algebras: an algebra $\langle A, F\rangle$ of expressions and an algebra $\langle B, G\rangle$ of meanings. The algebra of expressions is defined as an absolutely free algebra, cf. Sect. 1.4, with the finite lexicon of this language as its base set. Other than requiring that it be an absolutely free algebra, not many specific constraints on the algebraic structure of meanings were given by Montague. A first step in defining a structure of the semantic algebra was given in [Keenan, Faltz 1978; Keenan, Faltz 1985]. Keenan & Faltz start from the observation that the Boolean operators *and, or, not* are not limited to sentences as a category of operands, but can have operands of a variety of other syntactic categories like, e.g., common nouns, noun phrases, determiners, verb phrases, transitive verb phrases, adjective phrases, prepositions, prepositional phrases, and adverbial phrases. This led to a characterization of each of these categories as a Boolean algebra, i.e. as sets of objects closed under the Boolean operations. Under this approach, there is no algebra corresponding to the language as a whole in the sense of standard algebraizations for languages of logic. In particular, some categories are interpreted as homomorphisms, i.e. as mapping from one algebra to another algebra. A much more straightforward approach to an algebraization of a formal language was achieved in [Suppes 1976]. In the following, our focus will be on this approach.

## 15.2 Relational Grammar

In this section, we give an introduction to relational grammar in the context of model theory.

### Denoting grammar

There is a tradition in linguistics to represent the structure of an expression by a tree. The tree for expression (15.2), for instance, would appear as in Fig. 15.1. This tree represents two kinds of information about (15.2): the grouping of its constituents and the categorization of its constituents. The tree tells us that *European* is a classificatory adjective ($CA$), that *men* and *women* are common nouns ($N$), and that *and* is a conjunction ($Conj$). Moreover, it tells us that the

substring *men and women* forms a constituent, i.e. that the constituents *men*, *and*, and *women* belong closer together than *European* and *men*. Constituents of this kind are conventionally called noun phrases ($NP$).

A tree for an expression of a given language is conventionally derived by a grammar for that language. This is usually a so-called context-free grammar. The tree is therefore called a *derivation tree*. A context-free grammar which is able to derive the tree in Fig. 15.1 is the following one:

$$\begin{array}{rrcl} i) & NP & \rightarrow & CA + NP \\ ii) & NP & \rightarrow & NP + Conj + NP' \\ iii) & NP & \rightarrow & N \\ iv) & N & \rightarrow & men \\ v) & N & \rightarrow & women \\ vi) & CA & \rightarrow & European \\ vii) & Conj & \rightarrow & and \end{array} \qquad (15.7)$$

The symbol $\rightarrow$ means that any symbol of the left-hand side can be replaced by a symbol of the right-hand side. $NP$ is called the *start* symbol of the grammar, *men*, *women*, *European*, and *and* are called *terminal* symbols, $N$, $CA$, and $Conj$ as *non-terminal* symbols. The important rules are $i$) and $ii$), which allow any phrase of category $NP$ to be replaced by either an adjective and a phrase of category $NP$, or by two strings of the same category $NP$ combined by a conjunction of category $Conj$. They are important because they are recursive, i.e. any result of their application can be used as input to a new application.

A phrase structure grammar like (15.7) accounts only for the shape of a string of words but not for its meaning. Meanings are taken care of by *model theory*. Model theory provides semantic interpretations for a language with respect to a *model structure*. A model structure for a language is a pair of some non-empty set $D$ and a mapping $v$ from the terminal symbols of that language. $D$ is called the *domain* of the model structure. The mapping $v$ is called the *valuation function* of the model structure. The semantic interpretation assigns to each well-formed expression of the language a certain object of the model structure. This object is called the *denotation* of that expression with respect to that model structure. The stock of denotations is determined by the *hierarchy* over the model-structure.

If we assume that each denoting word of the string has a denotation we would then like to know the denotation for the whole string. The composition of meanings is effected by *semantic functions*. Semantic functions are therefore attached to the grammar rules. One way to attach semantic rules to our grammar (15.7) could be:

$$\begin{array}{rrcl} i) & [NP] & = & [CA] \cap [NP] \\ ii) & [NP] & = & [NP] \cup [NP'] \\ iii) & [NP] & = & [N] \\ iv) & [N] & = & [men] \\ v) & [N] & = & [women] \\ vi) & [CA] & = & [European] \end{array} \qquad (15.8)$$

The semantic functions are set-theoretical intersection in Rule i), set-theoretical union in Rule ii), and identity in the Rules iii) through vi). There is no semantic

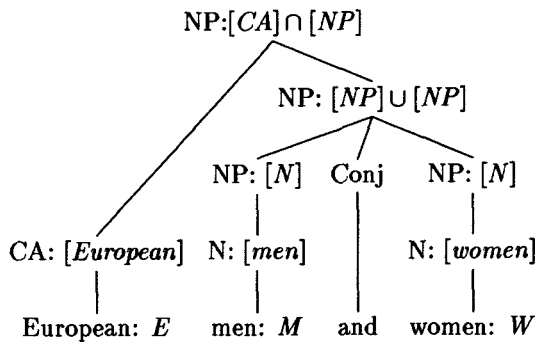



Fig. 15.2 Semantic tree for *European men and women*

function attached to Rule vii). The reason for this is that the word introduced by this rule does not have a denotation.

A phrase structure grammar with semantic functions attached to its production rules is called a *denoting grammar*. The notion of a denoting grammar was proposed in [Suppes 1973b]. Our denoting grammar ((15.7), (15.8)) allows us to compute the denotation of the expression (15.2) from its component denotations for every model structure of the grammar. As domain $D$ we may adopt the set of persons living on this planet in 1996. A quite different domain might have been picked by Boole in 1854. The grammar of (15.7) has the terminal symbols *men*, *women*, *European*, and *and*. We may then pick subsets of our domain $D$ as valuations for the words *men*, *women*, and *European*, namely the set of all male persons as valuation for *men*, the set of all female persons as valuation for *women* and the set of all citizens of some European country for *European*. No set is picked as valuation for the word *and*. The hierarchy used for the grammar ((15.7), (15.8)) is therefore any subset of

$$\mathcal{P}(D) \tag{15.9}$$

that is closed with respect to Boolean operations.

The computation of compound expressions can be represented by a *semantic tree*. The semantic tree for (15.2) is given in Fig. 15.2. If a leaf of this tree is labelled by a denoting word it will be assigned a denotation by the valuation function of the model structure under consideration. Any mother gets its denotation as the result of a certain semantic operation applied to the denotations of the respective daughters. This operation is determined by the particular rule by which the daughters are produced. The procedure is iterated down to the root of the tree. Since the root dominates every leaf of the tree, the root denotation is the denotation of the entire expression.

It should now be straightforward to write down a denoting grammar for the set of categorical sentences of (15.5). If this grammar contains the denoting grammar ((15.7), (15.8)) as a subgrammar it will derive categorical sentences with complex noun phrases like, e.g., *No European women are European men.*

## Relational extension

The hierarchy (15.9) provides denotations for expressions involving absolute terms like, e.g., *dog* or *pregnant*. Not all terms of a natural language are absolute though. There is a large and important class of relative terms. Standard examples are kinship terms like, e.g., *sister* or *uncle*. Another class of examples are spatial terms like, e.g., *in*, *above*, *behind*. There are terms involving some order like, e.g., *larger* or *louder*. Although there are ternary relations and relations of even higher degree in natural language, our focus here will be on binary relations only.

The denotation of a binary relative term is a binary relation on $D$. The relative noun *brother* denotes the set of pairs $\langle a, b\rangle$ of objects of $D$ such that $a$ is a brother of $b$, the preposition *in* denotes the set of pairs $\langle a, b\rangle$ of objects of $D$ such that $a$ is in $b$, the adjective *larger* denotes the set of pairs $\langle a, b\rangle$ such that $a$ is larger than $b$, and the transitive verb *hit* denotes the set of pairs $\langle a, b\rangle$ such that $a$ hits $b$. It is therefore necessary to replace the set (15.9) by

$$\mathcal{P}(D) \cup \mathcal{P}(D \times D) \tag{15.10}$$

But this is not sufficient. Although we will now be able to derive a denotation for, e.g., *brothers and sisters*, no denotation can be derived for *John is a brother of Mary*, or *London is near Paris*. What is needed are operations that compute sets or binary relations from sets and binary relations. For a list and definition of such operations see Sect. 1.2. In the following we shall define some operations and illustrate them by grammatical constructions from the English language.

## Image

The operation *image set* is defined in Sect. 1.2. From this we can derive the term

$$\bigcup_{x \in A} R(x) \tag{15.11}$$

and the term

$$\bigcap_{x \in A} R(x). \tag{15.12}$$

[Riguet 1948] proposed to use the term (15.11) to define a binary operation " taking a binary relation $R$ and a set $A$ as its arguments:

$$R"A = \bigcup_{x \in A} R(x). \tag{15.13}$$

This operation was called "coupe de première espèce" in [Riguet 1948]. It may be better known by the name *upper image* of set $A$ under relation $R$. Using the element relation the result of this operation amounts to the set

$$\{y | (\exists x)(x \in A \wedge xRy)\}. \tag{15.14}$$

This operation can be illustrated by a certain type of relative clause in which the relative pronoun has the role of the object. An example would be *who Mary likes*.

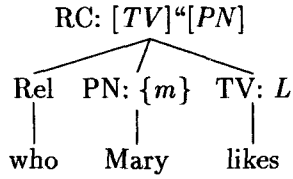
The denotation of this clause is the set of all those human beings of the domain $D$ such that Mary likes them, i.e. the set

$$\{y|mLy\} \ , \tag{15.15}$$

where $L$ is the binary relation denoted by the transitive verb *like* and $m$ is the individual denoted by *Mary*. Since

$$L\text{“}\{m\} = \{y|(\exists x)(x = m \wedge xLy)\}, \tag{15.16}$$

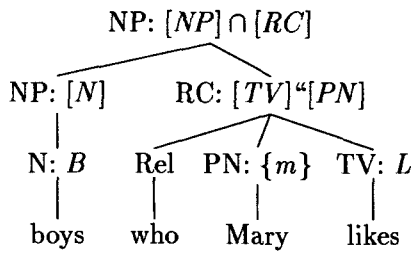
we are able to derive the denotation for the string *who Mary likes* from the denotations for *likes* and *Mary* by the operation defined in (15.13). The semantic tree for the string *who Mary likes* would then look as follows




(15.17)

where $RC$ = relative clause, $Rel$ = relative pronoun, $PN$ = proper noun, and $TV$ = transitive verb.

In Sect. 15.2 we have seen that nouns can be modified by property expressions. The semantic operation corresponding to modification is intersection. Intersection is therefore the operation that we have used in (15.8i). In (15.8i) properties were exhibited by adjectives. But adjectives are not the only syntactic category that may express a property. A property can also be expressed by a relative clause like, e.g., *who Mary likes*. An example would be *boys who Mary likes* where the noun *boys* is modified by the relative clause *who Mary likes*. The corresponding semantic tree would be:




(15.18)

In a fashion completely parallel to (15.13) [Riguet 1948] also proposed to use the term (15.12) to define the operation

$$R[A] = \bigcap_{x \in A} R(x) \tag{15.19}$$

called “coupe de deuxième espèce”. In terms of the element relation the result of this operation can be expressed by

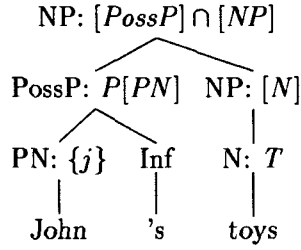
$$\{y|(\forall x)(x \in A \rightarrow xRy)\} \ . \tag{15.20}$$

A nice illustration of the operation defined in (15.19) is the possessive case in English like, e.g., *Mary's* in *Mary's toys*. The possessive case marker *'s* can be

thought of as denoting a binary relation $P$ relating human beings with objects like, e.g., Mary with a set of toys. The denotation of *Mary's* can then be derived from the singleton $\{m\}$ and the binary relation $P$ by

$$P[\{m\}] = \{y|(\forall x)(x \in \{m\} \rightarrow xPy)\} \ . \tag{15.21}$$

Common nouns can also be modified by possessive nouns like, e.g., *John's* in *John's toys*. Let *John's* denote the set of objects that belong to John. This set can be used to restrict the denotation of the noun phrase *toys*, i.e. the set $T$ of toys. The denotation of the noun phrase *John's toys* can then be derived along the following semantic tree:




(15.22)

We are now in a position to prove certain semantic facts about English. First, the phrases *John's toys and Mary's toys* and *John's and Mary's toys* are equivalent, since

$$[\textit{John's toys and Mary's toys}] = (P[\{j\}] \cap T) \cup (P[\{m\}] \cap T). \tag{15.23}$$

The equivalence of these expressions is an immediate instance of the distributive law of Boolean algebra. Second, the phrases *John and Mary's toys* and *John's and Mary's toys* are not equivalent. In order to see this recall from Rule ii) of (15.8) that the denotation of a phrase of two nouns combined by *and* is the union of the constituent denotations. We therefore have

$$[\textit{John and Mary}] = \{j\} \cup \{m\} \tag{15.24}$$

and, hence,

$$[\textit{John and Mary's toys}] = P[\{j\} \cup \{m\}] \cap T. \tag{15.25}$$

By the same token, we have

$$[\textit{John's and Mary's toys}] = (P[\{j\}] \cup P[\{m\}]) \cap T. \tag{15.26}$$

To show that the left hand sides of (15.25) and (15.26) are not equivalent it is sufficient to show that the respective right hand sides are not equivalent. Let us therefore assume that the set of toys $T$ has a teddy bear $t$ and a toy car $c$ as its elements:

$$T = \{t, c\} \tag{15.27}$$

Let the teddy bear belong to both John and Mary and let the toy car belong only to John, i.e. our relation $P$ of possession is defined

$$P = \{\langle j, t\rangle, \langle m, t\rangle, \langle j, c\rangle\}. \tag{15.28}$$

We then have

$$P[\{j\} \cup \{m\}] \cap T = \{t\} \tag{15.29}$$

and

$$(P[\{j\}] \cup P[\{m\}]) \cap T = \{t, c\} \cup \{t\} = \{t, c\} \tag{15.30}$$

Since $\{t\} \neq \{t, c\}$ it follows that *John and Mary's toys* and *John's and Mary's toys* have different meanings.

## Peirce product

Recall from Sect. 1.2 the definition of *Peirce product*

$$R{:}A = \{x \in D | (\exists y)(y \in A \wedge xRy)\}. \tag{15.31}$$

Let $O$ be the binary relation denoted by the transitive verb *own* and let $H$ be the set of houses denoted by the noun *house*. Instantiating (15.31), we get

$$O{:}H = \{x \in D | (\exists y)(y \in H \wedge xOy)\}. \tag{15.32}$$

Therefore, we have

$$x \in O{:}H \qquad \text{iff} \qquad (\exists y)(y \in H \wedge xOy). \tag{15.33}$$

The right hand side is true just in case there is a house that $x$ owns. The set $O{:}H$ can therefore serve as the denotation for the phrase *own some houses*.

Since the verb phrases *own some houses* and *own no houses* are contradictory, the respective denotations should be complements of each other:

$$[\textit{own no houses}] = \overline{O{:}H} \tag{15.34}$$

We thus have defined a semantic operation corresponding to the negative quantifier in object position.

Since any owner of no houses is identical to a not-owner of all houses we have

$$[\textit{own no houses}] = [\textit{not-own all houses}] \tag{15.35}$$

From the previous two equations we derive the equation

$$[\textit{not-own all houses}] = \overline{O{:}H} \ . \tag{15.36}$$

Replacing simultaneously the binary relation $O$ by its complement relation $\overline{O}$ and *not-own* by its complement *own* yields

$$[\textit{own all houses}] = \overline{\overline{O}{:}H} \ . \tag{15.37}$$

We thus have defined a semantic operation corresponding to the universal quantifier in object position by

$$[\textit{own all houses}] = \overline{\overline{O}{:}H}. \tag{15.38}$$

## Converse

The *converse* operation has the effect of turning around the order of the elements of a relation:

$$R^{\smile} = \{\langle y, x\rangle | \langle x, y\rangle \in R\} \tag{15.39}$$

As had already been observed by Peirce the standard use of this operation is to derive the meaning of a passive verb phrase from its active counterpart. Thus, if $I$ is the binary relation denoted by the transitive verb *invite* then $I^\smile$ is the binary relation denoted by the passivized verb *is invited by.*

The denotation for the verb phrase *invited by some philosophers* is

$$I^\smile : P \tag{15.40}$$

where $I$ is the binary relation denoted by *invite* and $P$ is the set denoted by *philosophers.*

**Theorem 15.2.1** For any subsets $A$, $B$ of some set and any binary relation $R$ on that set the following holds: If $B \cap \overline{(\overline{R^\smile} : A)} \neq \emptyset$ then $A \subseteq R : B$.

The validity of the argument

$$\frac{\textit{Some novels are liked by all people}}{\textit{All people like some novels}} \tag{15.41}$$

follows from Theorem 15.2.1. The theorem cannot be strengthened to a biconditional. Therefore the reverse argument

$$\frac{\textit{All people like some novels}}{\textit{Some novels are liked by all people}} \tag{15.42}$$

is not valid. So premise and conclusion do not have equivalent denotations. This is an interesting result, since it falsifies the widespread belief that passive sentences are synonymous with their active counterparts.

## Lower image

The operation

$$R \ulcorner A = \overline{R : \overline{A}} \tag{15.43}$$

was introduced in [Suppes, Zanotti 1977] by the name of *lower image* of set $A$ under relation $R$. This operation can be illustrated by a certain type of relative clause with the relative pronoun playing the role of an attribute like, e.g., *composers all of whose sons are composers.* The semantic tree for this expression is in Fig. 15.3 with $RC$ = relative clause, $UQ$ = universal quantifier, $RelPoss$ = relative possessive pronoun, $RN$ = relative noun, $Cop$ = copula, $VP$ = verb phrase, and $P$ = preposition. The denotations involved are the set $C$ of composers and the binary relation $S$ of being a son of.

An element of the root denotation of the semantic tree in Fig. 15.3 is Leopold Mozart (at least in case we restrict our domain to adult people thus excluding Wolfgang Amadeus' five siblings who died at infant age). Although Johann Sebastian Bach is known to have at least four sons who were also composers, he does not belong in this class for the reason that he had seven more sons who are not known to have become composers themselves. On the other hand, George Frederick Handel and Franz Schubert certainly belong in this class although they

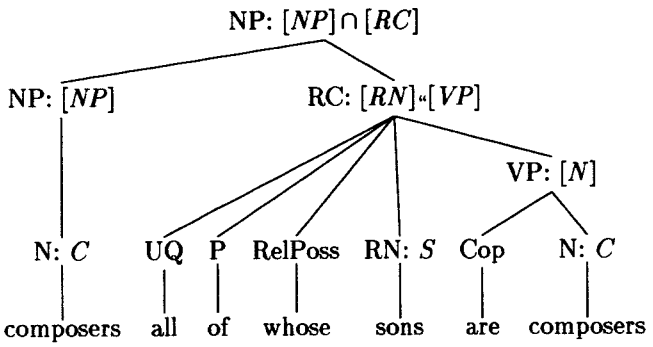



Fig. 15.3 Semantic tree with relative possessive pronoun

are not known to have had any children at all. This is odd but in line with the conventional interpretation of universal quantification in modern logic.

Notice that the relative clause denotation makes essential use of the lower image operation. Since by definition $S_{\cdot}C = \overline{S^{\cdot}\overline{C}}$ and, since

$$\overline{S^{\cdot}\overline{C}} = \{x|(\forall y)[ySx \rightarrow y \in C]\}$$

we can see that this set reflects exactly our intuitive understanding of the phrase.

## Restriction

So far we have only considered the modification of absolute terms. But there is also modification of relative terms. For instance, in kinship terminology the noun *brother* is defined by the phrase *male sibling*. The head of this phrase, which is *sibling*, is a relative term but the modifier of this phrase, which is *male*, is absolute. Therefore *sibling* denotes a binary relation and *male* denotes a set. But sets and binary relations cannot be intersected. The operation that is suitable for this case is *domain restriction* of a binary relation $R$ by a set $A$ defined as follows:

$$R \restriction A = R \cap (A \times D). \tag{15.44}$$

The relation $B$ denoted by *brother* can then be defined

$$B = S \restriction M, \tag{15.45}$$

where $S$ is the binary relation denoted by *sibling* and $M$ is the set denoted by *male*. This operation provides the correct definition for brother: $x$ is a brother of $y$ iff $x$ is a male sibling of $y$, and $x$ is a male sibling of $y$ iff $x$ is a sibling of $y$ and $x$ is male.

An analogous operation of *range restriction* of $R$ by $A$ can be defined as follows:

$$R \upharpoonright A = R \cap (D \times A) \tag{15.46}$$

## Initial segment

In (15.8*i*) a semantic function was given for combinations of a classifying adjective with a noun phrase. Not all adjectives are classifying though. Consider examples

like, e.g., *old*, *high*, or *ugly*. These adjectives differ from classifying adjectives by deriving comparatives like *older than*, *higher than*, or *uglier than*. We therefore call them *comparison adjectives* or *intensive adjectives* (*IA*). Since comparison implies order, a comparison adjective denotes an ordering relation on $D$. But if a comparison adjective like, e.g., *high* denotes an ordering relation, then an operation is called for to derive the meaning of *high mountain* from the denotations for *high* and *mountain*. The notion that turns out to be useful here is the notion of an initial segment of a binary relation. Let $A$ be some set, $R$ some ordering (strict partial) relation $R$ on $A$ and $c$ an element of $A$. Then the *initial segment* of $A$ with respect to $R$ and criterion object $c$ is defined

$$IS(R, A, \{c\}) = (R : \{c\}) \cap A. \tag{15.47}$$

So the denotation of the phrase *high mountain* will be

$$(H : \{m\}) \cap M \tag{15.48}$$

where $H$ is the relation denoted by *high*, $M$ is the set denoted by *mountain* and $m$ is some element of $M$. If a mountain is higher than this criterion mountain $m$ (or at least as high as it) we call it a high mountain and if it is lower than it we do not call it a high mountain.

This solution nicely accounts for the fact that the order of adjectives modifying a noun sometimes matters. The expressions *Dutch high mountains* and *high Dutch mountains* may have different denotations.

## 15.3 Further Refinements

In addition to the operations mentioned in the preceding section there are some operations of a very fundamental kind that have not been mentioned yet. These operations are *composition*, *identity*, and *domain*. The reason why they have not been mentioned is that they cannot be given a direct natural language interpretation. This is all the more surprising in the case of the operations of composition and identity. They will, however, be used to define more complicated operations for which very natural interpretations exist. We would like to mention here only the operations *Ref*, *Rec*, *Poss*, *RecPoss*, and *Id*. They are tailored specifically to the needs of certain natural language constructions. *Poss* is introduced as a counterpart for the possessive pronoun construction, *Ref* is intended to construe the semantics of verb phrases with a reflexive pronoun, *Rec* is introduced as a counterpart for the reciprocal pronoun construction, *RecPoss* is introduced as a counterpart for the reciprocal in the possessive case, *Id* is introduced as a counterpart for the identity pronoun construction.

### Poss

From Sect. 1.2 we know that the operation *domR* computes the set of all elements of $D$ related by $R$ to some element of $D$. From the preceding section recall the definition of range restriction of a binary relation by a set. Let $R$ be an arbitrary

relation over $D$ and let $A$ be an arbitrary subset of $D$. Let $P$ be the relation of possession in all models. We are then in a position to derive the term

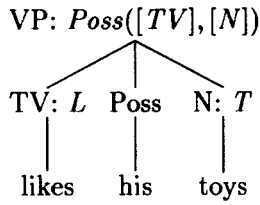
$$Poss(R, A) = \overline{dom((P \restriction A) \cap \overline{R})}. \tag{15.49}$$

The operation *Poss* returns a subset of the universe $D$ as its value.

We have already considered possessive phrases in Sect. 15.2. Of more interest is the question of how possessive anaphoric pronouns are interpreted like in, e.g.,

$$\textit{John likes his toys} \tag{15.50}$$

where the interpretation of the possessive pronoun *his* depends on the interpretation of the subject term of the sentence. For the verb phrase *likes his toys* we propose the tree




(15.51)

According to this tree the denotation of the verb phrase is

$$\overline{dom((P \restriction T) \cap \overline{L})} \tag{15.52}$$

where $L$ is the denotation for the transitive verb *likes* and $T$ is the denotation for the common noun *toys*. That (15.52) corresponds to the intuitive meaning of the verb phrase follows from the fact that

$$x \in \overline{dom((P \restriction T) \cap \overline{L})} \tag{15.53}$$

is true just in case the following condition holds:

$$(\forall y)[[T(y) \textit{ and } P(x, y)] \Rightarrow L(x, y)] \tag{15.54}$$

The trees for the phrases *like her toys* or *like their toys* would look very similar to (15.51). A proper treatment would have to introduce gender into the grammar.

One might be tempted to derive the interpretation of (15.50) by replacing *his toys* by *John's toys*. But that would be a mistake. For the sentence *John likes John's toys* could not be generalized to account for the contextual dependence of the interpretation of the possessive pronoun on the referent of the subject term. This can be seen if the subject term is replaced by a complex term like in, e.g., *John and Mary like their toys*, which is not equivalent to the sentence *John and Mary like John and Mary's toys*.

## Ref

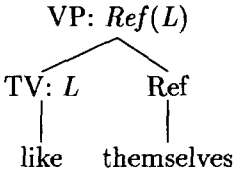
In Sect. 1.2 the notion of an identity relation $I_X$ is defined. Let $X = D$, i.e. let us consider the "full" identity $I_D$ over $D$. With the help of intersection, the full identity relation, and the domain operation we can derive the term

$$dom(R \cap I_D). \tag{15.55}$$

Let us refer to this operation by the name *Ref* for the reason that it corresponds to expressions with a reflexive pronoun like, e.g.

*John and Mary like themselves.* (15.56)

The semantic tree for the verb phrase of this sentence looks like this:




(15.57)

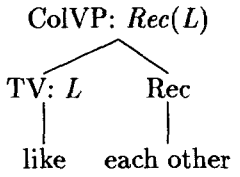
That (15.57) corresponds to the intuitive meaning of the verb phrase follows from the fact that $x \in dom(R \cap I_D)$ iff $xRx$.

## Rec

Let $R$ be an arbitrary relation on domain $D$. The operation $R \cap R^{\smile}$ is then the largest symmetric subrelation of $R$. For reasons that we shall not go into here we prefer to define an operation *Rec* by

$$Rec(R) = (R \cap R^{\smile}) - I \tag{15.58}$$

rather than simply by $R \cap R^{\smile}$. This operation can be illustrated by the reciprocal pronoun construction. An example would be *John and Mary like each other.* The semantic tree for the verb phrase of this sentence is




(15.59)

The structure of the sentence *John and Mary like each other* is quite similar to the structure of the sentence *John and Mary like themselves*: a compound subject term is followed by a verb phrase that consists of a transitive verb and some pronoun as object term. However, the semantic tree (15.59) is quite different: The predicate term in (15.57) has the label *VP* but the predicate term in (15.59) has the label *ColVP*. The purpose of different categories is to announce a difference in semantic type: Whereas an expression of category *VP* denotes a set, an expression of category *ColVP* denotes a binary relation.

**Theorem 15.3.1** *Rec*$(R)$ is a symmetrical relation.

This theorem guarantees the validity of the following argument:

$$\frac{\textit{John and Mary like each other}}{\textit{Mary and John like each other}} \tag{15.60}$$

The validity of the argument

$$\frac{\textit{John and Mary like each other}}{\textit{John likes Mary}} \tag{15.61}$$

follows from the fact

$$\text{If } (A \times B) \cap R \neq \emptyset \text{ then } A \cap (R : B) \neq \emptyset,$$

since $((A \times B) \cap R) : B \subseteq ((A \times B) : B) \cap (R : B) = A \cap (R : B)$.

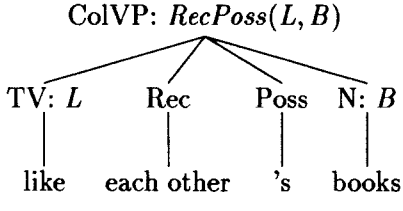
## RecPoss

With the composition of two relations $R$ and $S$ introduced in Sect. 1.2 we define

$$\mathit{RecPoss}(R, A) = \overline{\overline{R};I_A;P^\smile} \cap \overline{P;I_A;\overline{R^\smile}} \cap \overline{I} \tag{15.62}$$

The operation *RecPoss* takes a binary relation and a set as its arguments and returns a binary relation.

This operation is designed for reciprocal pronoun occurring in the possessive case like in, e.g., *John and Mary like each other's books.* The semantic tree for the verb phrase *like each other's books* would look as follows:




(15.63)

It has the denotation

$$\overline{\overline{L};I_B;P^\smile} \cap \overline{P;I_B;\overline{L^\smile}} \cap \overline{I}. \tag{15.64}$$

This is what it is supposed to denote, since

$$\langle x, y\rangle \in \overline{\overline{L};I_B;P^\smile} \cap \overline{P;I_B;\overline{L^\smile}}$$

is equivalent to

$$(\forall z)(Bx \rightarrow [yPz \rightarrow xLz] \wedge [xPz \rightarrow yLx]])$$

which is what the verb phrase *like each other's books* intuitively amounts to.

**Theorem 15.3.2** *RecPoss*$(R, A)$ is a symmetric relation for arbitrary sets $A$ and relations $R$.

An immediate consequence of this theorem is that the argument

$$\frac{\textit{John and Mary like each other's books}}{\textit{Mary and John like each other's books}} \tag{15.65}$$

is valid.

### Id

*Id* is an operation that turns a relation and a set into a relation. It is defined

$$Id(R, A) = \overline{(R;I_A;\overline{R^\smile})} \cap \overline{(\overline{R};I_A;R^\smile)} \tag{15.66}$$

It can be illustrated by the expression *same* like occurring in

*John and Mary read the same books* (15.67)

Notice that the verb phrase *read the same books* of this semantic tree denotes a binary relation. The sentence (15.67) can be paraphrased by *John reads the same books as Mary*, which brings out better the feature that the sentence is relational. According to our analysis the denotation of the verb phrase *read the same books* is $Id(R, B)$. That $Id(R, B)$ indeed exhibits the meaning denoted by (15.67) can be seen from the fact that

$$\langle x, y\rangle \in \overline{R;I_B;\overline{R^\smile}} \cap \overline{\overline{R};I_B;R^\smile}$$

holds just in case $(\forall z)(Bz \rightarrow (xRz \leftrightarrow yRz))$ is true.

**Theorem 15.3.3** $Id(R, A)$ is symmetric.

**Theorem 15.3.4** $Id(R, A)$ is transitive.

For the definitions of the notions of symmetry and transitivity of a binary relation see Sect. 1.2. From 15.3.3 and 15.3.4 it follows that $Id(R, B)$ is an equivalence relation on the set $R : B$ of book readers. But if $Id(R, B)$ is an equivalence relation then the argument

$$\frac{\begin{array}{l}\textit{John and Mary read the same books}\\ \textit{Bill and Mary read the same books}\end{array}}{\textit{John and Bill read the same books}}$$

should be valid, which is indeed the case.

## 15.4 Procedural Semantics

So far our denotations for English words have been given in the fashion of standard model theory. This view is completely static: a world is conceived of as a set of objects with certain properties and relations. Applied to actions this view leads to certain absurdities. Consider the action described by the sentence *John adds 34711 and 9263*. Under this approach the verb *add* denotes some binary relation between a human individual and a sequence of numbers:

$$\{\langle j, \langle 34711, 9263\rangle\rangle\} \tag{15.68}$$

This analysis does not account for the fact that adding numbers is an operation that returns some result. But an account of successful addition should return the result of the addition. It has therefore been proposed to define natural language meanings in terms of procedures.

We now interpret the elements of the Boolean algebra as sets of states (of some agent) and the elements of the relation algebra as sets of state-transitions.

Following a suggestion of [Suppes 1973b], we view the agent as a machine that is able

- to move into four directions (left, right, up, down) in a gridlike environment
- to retain representations of perceived objects
- to recognize objects as objects of a certain kind, e.g. numbers.

Let us also assume that this machine has two registers. In line with [Suppes 1973a], we let the machine have a focus register F to keep track of the current location of the agent in the environment and a memory register M to keep track of objects that should be kept in memory for some time. Any state of the agent can then be represented by an ordered pair: the first component representing an object in visual focus and the second component representing an object in memory.

For purposes of illustration we use the case of arithmetical instructions like, e.g., column addition that require an agent to be able to identify objects like digits arranged in rows and columns like this

$$\begin{array}{ccc} & 3 & 4 \\ 1 & 7 & 8 \\ & & 7 \\ & 5 & 5 \\ - & - & - \end{array} \tag{15.69}$$

It should be noted that the set of symbols is not just the set of digits 0,1,2,... together with a bar and a symbol $b$ for blanks, but rather a set of occurrences or tokens of those symbols: the symbol 7 of the second row has to be distinguished from the symbol 7 in the third row. In the same way, empty spaces have to be kept distinct. The perceptual environment for arithmetic instruction (15.69) can then be represented as follows:

$$\begin{array}{ccc} b_1 & 3 & 4 \\ 1 & 7_1 & 8 \\ b_2 & b_3 & 7_2 \\ b_4 & 5_1 & 5_2 \\ -_1 & -_2 & -_3 \\ b_5 & b_6 & b_7 \end{array} \tag{15.70}$$

Any arrangement of this kind can be identified as a finite geometry of two relations $V$ ("vertically below") and $H$ ("horizontally left of"), in the sense of [Crangle, Suppes 1987]. So we have $\langle 7,4\rangle \in V$ but $\langle 1,4\rangle \notin V$. Both $V$ and $H$ are strict ordering relations, cf. Sect. 1.2 We refer to 3 as the *left neighbour* of 4, to 8 as the *lower neighbour* of 4, to elements b,3,4 as the *V-maximal*, and to elements $-_1$, $-_2$, $-_3$ as the *minimal* elements of $V$.

Assume that the agent is located on the top square of the middle column. Assume further that the agent is in an initial state. Then the state of the agent would be $\langle 3,\varepsilon\rangle$ where $\varepsilon$ indicates that the register is empty.

Assume now that our agent is given the command *Look!* The procedure expected of the agent would be that the content of the F register is changed to any

other field, i.e. our agent would now be in state $\langle x, \varepsilon \rangle$. The exact destination of looking is given by additional wording. So the meaning of *look* has brought about the following state transition: $\langle \langle 3, \varepsilon \rangle, \langle x, \varepsilon \rangle \rangle$ It is then plausible to see that the meaning of the verb *look* can be identified with the set

$$\{\langle \langle f, m \rangle, \langle f', m \rangle \rangle\}$$

of state transitions.

Register M is the register that stores the content of the agent's memory. Let us assume that the agent has perceived the symbol in the first row and then gets the command to remember that number, then the agent's state would get changed to $\langle 3, 3 \rangle$. So the procedure associated with the verb *remember* has effected the following state transition: $\langle \langle 3, \varepsilon \rangle, \langle 3, 3 \rangle \rangle$ Again it is easy to see that *remember* can be identified with the following transition set:

$$\{\langle \langle f, m \rangle, \langle f, f \rangle \rangle\}$$

A minimal movement down can be defined by the following transition set:

$$\{\langle \langle f, m \rangle, \langle f', m \rangle \rangle | f' V f \text{ and } f' \text{ is a neighbour of } f\}$$

We let this set be the denotation for the adverb *down*.

A maximal movement up can be defined by the following transition set:

$$\{\langle \langle f, m \rangle, \langle f', m \rangle \rangle | f V f' \text{ and } f' \text{ is maximal}\}$$

We use this as the denotation for *top*.

An arbitrary movement in both up and down directions can be defined by the following transition set:

$$\{\langle \langle f, m \rangle, \langle f', m \rangle \rangle | f V f' \vee f' V f\}$$

We adopt this set as our denotation for the noun *column*.

Since the denotations of *top* and *column* are relational, we keep track of this by assigning these words relational categories: *top* the category $RA$ (= relational adjective) or $RN$ (= relational noun) and *column* the category $RN$. The relational character distinguishes both *top* and *column* from a noun like *number* that denotes the set of all states with a number symbol, i.e. a digit, in its first component:

$$\{\langle f, m \rangle : f \text{ is a number}\}$$

An overview of our toy lexicon is given in Table 15.1: This lexicon could easily be extended by adding corresponding entries for *bottom*, *up*, *left*, *right*, *leftmost*, *rightmost*, *bar* etc., for details, see [Böttner 1992a].

## Composition

To derive non-primitive procedures from our lexical procedures we use the familiar operations from relation algebra. Composition of two relations $R$ and $S$ is no doubt the most important operation of relational algebra. It is defined as follows:

$$R;S = \{\langle x, y \rangle | (\exists z)(xRz \wedge zSy)\} \tag{15.71}$$

| | Category | Denotation |
|---|---|---|
| look | $V$ | $\{\langle\langle f,m\rangle,\langle f',m\rangle\rangle\}$ |
| remember | $V$ | $\{\langle\langle f,m\rangle,\langle f,f\rangle\rangle\}$ |
| top | $RN, RA$ | $\{\langle\langle f,m\rangle,\langle f',m\rangle\rangle \mid fVf'$ and $f$ $V$-maximal$\}$ |
| down | $Adv$ | $\{\langle\langle f,m\rangle,\langle f',m\rangle\rangle \mid f'Vf$ and $f'$ neighbour of $f\}$ |
| column | $RN$ | $\{\langle\langle f,m\rangle,\langle\langle f',m\rangle\rangle \mid fVf' \vee f'Vf\}$ |
| number | $N$ | $\{\langle f,m\rangle \mid f$ *is a number*$\}$ |
| spot | $N$ | $\{\langle f,m\rangle \mid f \neq \varepsilon\}$ |

Table 15.1 Procedural Lexicon

If $R$ and $S$ are procedures the intuitive meaning of this operation is the sequencing of two procedures: first doing $R$ and then $S$. If the two relations $R$ and $S$ are identical, i.e. if $R = S$ we may abbreviate $R;R$ by $R^2$. We define $R^0 = I$.

An example of this composition is the denotation of the phrase *column to the left*:

$$[\textit{left}];[\textit{column}]$$

Intuitively, this procedure first shifts the agent's focus to the field left of the field of the current focus and then to every state with fields of the same column in focus.

## Transitive closure

This operation can be generalized to any finite number of operands and will then be called the $n$th *power* of $R$. The *transitive reflexive closure* $R^*$ of a binary relation $R$ is defined as follows:

$$R^* = \bigcup_{i \in I} R^i \tag{15.72}$$

The use of the closure operation is to define iteration operations. In the context of computation two types of iteration are conventionally distinguished: *while*-iteration and *until*-iteration. In *while*-iteration, the computation is triggered by a certain condition. In *until*-iteration the computation is started and continued until a certain condition is met to stop it. In [Böttner 1992a] we referred to these operations by symbols $\vdash$ and $\dashv$ with the understanding that the horizontal bar represents a process and the vertical bar represents a state. One should bear in mind though that in general there is no symmetry involved, i.e. $A \vdash R \neq R \dashv A$.

## While-Iteration

While-iteration is defined as a binary operation

$$A \vdash R = (R \restriction A)^*; (I \restriction \overline{A}) \tag{15.73}$$

This procedure can be described intuitively by the following steps:

1. Go to the top!
2. While you are on a field with no number in it, go one field down!

The operation can be illustrated by the phrase *top number*. It denotes the procedure

$$[top];(\overline{[number]} \vdash [down])$$

Assume the focus of the agent in environment (15.70) is on $b_4$. The procedure $[top]$ then takes the focus to $b_1$. Since no number in this field, the triggering condition of procedure $(\overline{[number]} \vdash [down])$ is met and the focus is shifted to the next field below the current field. Since there is a number in this field, the triggering condition of the procedure is no longer met and the procedure stops on the field with 1 in focus.

### Until-Iteration

The until iteration is defined as follows:

$$R \dashv A = R;(R \upharpoonright \overline{A})^*;(I \upharpoonright A) \tag{15.74}$$

The procedure denoted by this expression can be phrased by the instruction *Move down to the first square with a digit in it!*

An example of this operation is the expression *next number down*. It denotes the procedure

$$[down] \dashv [number] \tag{15.75}$$

If the agent's current focus is on $7_1$ in (15.70), this procedure will shift the focus to the field immediately below this field, which is $b_3$. Since this field has no number in it, the condition is met for iterating the procedure. After applying $[down]$ once more, the focus is shifted to $5_1$. Since this field has a number in it, the stopping condition is met and $[down]$ does not apply any more. The shift from $7_1$ to $5_1$ is exactly what one would expect our procedure to do.

As before we have not given explicit rules of grammar. A more elaborated grammar with a procedural semantics is given in [Böttner 1992a]. It derives semantic trees for instructions of considerable linguistic complexity like, e.g., *Look two numbers down! Look three spots to the left! Look down until you see a bar! Look at the top rightmost number! Write the ones digits of the number in the next space down! Continue looking down!* An example is the semantic tree for *Look at the top number!* represented in Figure 15.4.

## 15.5 Conclusion

In this chapter we have shown how relation algebra can be fruitfully applied to the semantics of natural language. We have described two approaches, a static one in the standard sense of model theory and a dynamic one in terms of states of an agent understanding natural language.

The construction of a relational grammar for some fragment of English appears to be straightforward and it appears that the approach can be applied to other natural languages as well. But things may not always be as easy as they may seem to be from the examples considered so far. This may become apparent if one tries to express the sentences

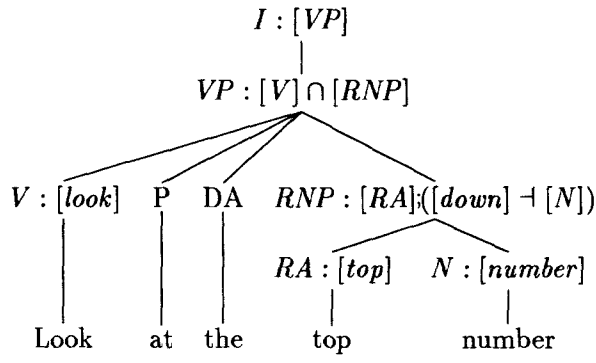



Fig. 15.4 Semantic Tree for *Look at the top number!*

i. *John bought five apples*
ii. *John is running under a tree*

in relation algebraic terms. Although i appears to be similar in structure to, e.g., *John bought some apples*, or *John bought red apples*, which both can be construed in terms of relational algebra, no simple solution along these lines can be given for cardinal expressions. The sentence ii has at least two different readings. According to one reading, the tree is the location of John's running. According to the other reading the tree is the destination of John's running. Whereas the locational meaning can easily be achieved in our model theory, it is not clear how the destinational meaning could be construed.

One major advantage of construing a natural language in terms of relational algebra is that natural languages can be viewed as equational languages (see Sect. 1.2). Equational languages are highly convenient for the purpose of computing inferences. Two caveats may be appropriate, though. First, the fragments of natural languages described by relational grammars are very small. Many important areas of natural language have not been dealt with. What is still missing, for instance, is the extension to relations of higher than binary degree, to mass nouns, or to adverbs. Also there has not yet been proposed a satisfactory treatment of tense. Second, relation algebra will certainly not be sufficient to account for all valid inferences in a natural language. A case in point is the semantics for place and time adverbials: they will require an analysis of space and time. For a first step into the analysis of space see [Crangle, Suppes 1989]. Intuitively, our world is made up of objects, persons, matter, colours, sounds, events, states, processes, actions. One might therefore have sincere doubts whether this abundant variety can be captured by an ontology only of sets and binary relations.