14

The Proper Treatment of Coordination in Peirce Grammar

Hans Leiss

Abstract

Peirce grammar provides a first-order, relational semantics for extensional fragments of natural language. Proposals in the literature to model natural language coordination by boolean operations in Peirce grammar lead to inadaequate results. We propose to equip Peirce grammar with an additional sort of finite trees and to admit recursion on trees for defining the evaluation of expressions. We demonstrate how this leads to a better treatment of coordination and other recursive constructions, such as the inverse linking of quantifiers in nested noun phrases.

Keywords Peirce grammar, relational semantics, coordination, inverse linking

14.1 Introduction

A Peirce grammar is a context-free grammar with an interpretation of expressions as sets or binary relations on a universe V. Peirce grammars have evolved from an interpretation of context-free grammars in extended relation algebras by Suppes (1976), which in turn owe much to the work of Peirce (1932) on relation algebra as a semantics of natural language.

While relational semantics is well-established for programming languages, it is less known in the context of natural language, despite its roots. Peirce grammars have been used by Suppes (1976) and Böttner (1999) to model fragments of English and German. The values of adjectives, nouns, prepositions and verbs are sets or relations, those of individual names are atomic sets, but function words like determiners

FG-MoL 2005.
James Rogers (ed.).

and conjunctions are treated syncategorematically. The value of complex expressions is computed bottom-up by algebraic terms attached to grammar rules.

A distinctive feature of Peirce grammar is its algebraic semantics: there is no infinite hierarchy of types, no variables and variable binders, no application of meaning objects to other meaning objects, but just construction of meanings from basic ones by applications of a few given operations; there are no built-in relations between meanings such as generalized quantifiers as relations between sets, except for equality and a definable subsumption relation. Since the algebraic operations have an equational axiomatization, equational reasoning is sufficient to establish semantic equivalence of expressions, which also makes Peirce grammar attractive from a computational point of view.

It has been claimed that this framework is too weak to cope with recursive syntactic structures and needs infinitely many categories or grammar rules to fix this (cf. Jäger (2004)). After introducing Peirce grammar in section 2, we address the problem of recursive structures in section 3, by adding finite trees to Peirce algebras and structural recursion to the evaluation. We demonstrate our extension on two examples: coordination of nouns and adjectives and inverse linking of quantifiers in nested noun phrases. Relations to other theories are discussed in section 4, and advantages, drawbacks and open problems in section 5.

14.2 Peirce grammar

A Peirce algebra is a two-sorted algebra of so-called *set*s and *relations* with the ususal boolean operations on each sort as well as some operations relating the two sorts. A Peirce grammar is a context-free grammar G with an evaluation function $[\![\cdot]\!] : L(G) \to \mathcal{P}$, assigning to each expression $w \in L(G)$ a meaning $[\![w]\!]$ in a Peirce algebra $\mathcal{P}$. In particular, the meaning objects are first-order objects; higher-order objects like functions in Montague-grammar are not admitted.

14.2.1 Peirce algebra

Definition 30 Let V be a set, called the *universe*. $\mathcal{P}_V = (\mathcal{B}_V, \mathcal{R}_V, :, ^c)$, the *Peirce algebra over* V, consists of

1. the boolean algebra $\mathcal{B}_V = (2^V, \cup, \cap, \bar{}, \emptyset, V)$ of all subsets of V,
2. the relation algebra $\mathcal{R}_V = (2^{V\times V}, \cup, \cap, \bar{}, \emptyset, V \times V, ;, ^\smile, I)$ of all binary relations on V, where $;, ^\smile, I$ are
   (a) the *relative product* $S;T := \{\langle a, c\rangle \mid \exists b(R(a, b) \wedge T(b, c))\}$,
   (b) the *converse* $S^\smile := \{\langle b, a\rangle \mid S(a, b)\}$, and
   (c) the *identity relation* $I := \{\langle a, a\rangle \mid a \in V\}$,

3. the *peircean product* : from $\mathcal{R}_V \times \mathcal{B}_V$ to $\mathcal{B}_V$, where
$$R : B := \{a \in V \mid \exists b \in V(R(a,b) \wedge B(b))\}$$
is the *inverse image* of the set B under the relation R, and
4. the (right) *cylindrification* $^c : \mathcal{B}_V \to \mathcal{R}_V$, where
$$A^c := \{\langle a,b\rangle \in V \times V \mid A(a) \wedge b \in V\} = A \times V$$
is the (right) *cylinder* over the set A.

An equational axiomatization of the properties of the operations of $\mathcal{P}_V$ by Brink et al. (1994) defines the notion of a Peirce algebra $\mathcal{P} = (\mathcal{B}, \mathcal{R}, :, ^c)$, where $\mathcal{B} = (B, +, \cdot, \bar{}, 0, 1)$ is a boolean algebra, $\mathcal{R} = (R, +, \cdot, \bar{}, 0, 1, ;, ^\smile, I)$ a relation algebra, and " : " : $R \times B \to B$ and $^c : B \to R$ are analogs of the Peirce product and cylindrification.

We will use *Peirce-terms* built from set- and relation variables, the constants $0, 1, I$ and the function symbols $+, \cdot, \bar{}, :, ^\smile$. Each term has a type, *set* or *rel*. The ambiguity of $0, 1, +, \cdot, \bar{}$ is easily resoved by the context, or indicated as in $+^{\mathcal{B}}$ versus $+^{\mathcal{R}}$, and likewise for the type of variables.

Many useful operations on sets and relations can be defined and their equational properties be proved in Peirce algebra. Some of these, with their interpretation in $\mathcal{P}_V$, are: the *relative sum* $\dagger$ of two relations,
$$R \dagger S := \overline{\overline{R}; \overline{S}} = \{\langle a,b\rangle \in V \times V \mid \forall c \in V(R(a,c) \vee S(c,b))\},$$
the *peircean sum* $\dagger$ of a relation and a set,
$$R \dagger B := \overline{\overline{R} : \overline{B}} = \{a \in V \mid \forall b \in V(R(a,b) \vee B(b))\},$$
the *domain* and *range* of a relation,
$$dom(R) := R : 1, \quad ran(R) := \breve{R} : 1 = \{b \in V \mid \exists a \in V\, R(a,b)\},$$
the *cartesian product* of two sets,
$$A \times B := A^c \cdot (B^c)^\smile = (A \times V) \cap (V \times B).$$

14.2.2 Peircean operations as natural language constructs

Any Peirce algebra term $t(x_1, \ldots, x_n)$ can be translated into a first-order formula $\varphi(X_1, \ldots, X_n, y, z)$ with binary relations X_i and two individual variables y, z, but not conversely: the quantification in these formulas is of a special form. Basically, Peirce (1932) observed that verbs and nouns combine via a quantifier that connects one argument of the former with one of the latter, as in
$$Qz(R(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n) \circ S(y_1, \ldots, y_{j-1}, z, y_j, \ldots, y_m)),$$
where Q is a quantifier and $\circ$ a binary boolean operator. Some examples with $n, m \leq 2$ are:

1. combining a transitive verb and a relational noun to a transitve verb:
   knows a brother of $= K\,;B = \{\langle a,c\rangle \mid \exists b(a\,K\,b \wedge b\,B\,c)\}$
   knows no brother of $= \overline{K\,;B} = \overline{K} \dagger \overline{B}$
   knows all brothers of $= K \dagger \overline{B} = \{\langle a,c\rangle \mid \forall b(b\,B\,c \to a\,K\,b)\}$
   knows not all brothers of $= \overline{K \dagger \overline{B}} = \overline{K}\,;B$
   knows only brothers of $= \overline{K} \dagger B = \{\langle a,c\rangle \mid \forall b(a\,K\,b \to b\,B\,c)\}$
   knows not only brothers of $= \overline{\overline{K} \dagger B} = K\,;\overline{B}$
2. combining two relational nouns to a relational noun:
   daughter of a brother of $= D\,;B$
   reader of all books by $= R \dagger \overline{B}$
3. combining a transitive verb and a common noun to an intransive verb:
   likes a book $= L : B = \{a \mid \exists b(L(a,b) \wedge B(b)\}$
   likes all books $= L \dagger : \overline{B} = \{a \mid \forall b(B(b) \to L(a,b))\}$

Moreover, verbs of equal subcategorization combine by boolean operations, and the converse $^\smile$ amounts to the passive of transitive verbs.

14.2.3 Peirce grammar

From now on, we extend the Peirce-operations by a few *test functions*, $E, U : \mathit{set} \to \mathit{set}$ and $\mathit{ER}, \mathit{UR} : \mathit{rel} \to \mathit{rel}$. They are to be interpreted as testing whether their argument is $\neq 0$, or $= 1$, and return one of the values 0 or 1 which serve as truth values *false* and *true*.

In a Peirce algebra $\mathcal{P} = (\mathcal{B}, \mathcal{P}, :, ^c)$, we identify the restrictions of $\mathcal{B}$ resp. $\mathcal{R}$ to its elements $\{0,1\}$ with the algebra $\mathbb{B}$ of truth values.

Definition 31 A *Peirce grammar* $PG = (G, \mathit{type}, e)$ consists of

1. a context-free grammar $G = (T, N, S, P)$, with finte sets of terminals T, nonterminals N, syntax rules $P \subseteq N \times (T \cup N)^*$, and a sentence categoy $S \in N$,
2. a function $\mathit{type} : N \cup T \to \{\mathit{set}, \mathit{rel}\}$, which may be partial on the set of terminals,
3. a mapping e, which assigns to each rule $r = (A \to A_1 \cdots A_k) \in P$ a Peirce-term $t_r(X_{i_1}, \ldots, X_{i_n})$ of type $\mathit{type}(A)$, where X_j has $\mathit{type}(A_j)$ when $A_{i_1} \cdots A_{i_n}$, $n \geq 1$, is the subsequence of typed elements of $A_1 \cdots A_k$.

Convention: Instead of $e(A \to \alpha) = t$ we write $A \to \alpha\,[t]$, using the typed terminals and nonterminals of α as variables in t. Several rules for the nonterminal A may be combined as in

$$A \to \alpha_1\,[t_1] \mid \cdots \mid \alpha_n\,[t_n].$$

Definition 32 An *interpretation* $(\mathcal{P}, v)$ of a Peirce grammar $(G, type, e)$ consists of a Peirce algebra $\mathcal{P} = (\mathcal{B}, \mathcal{R}, :, ^c, E, U, ER, UR)$ with test functions and an evaluation function $v : T \to \mathcal{P}$, assigning to each typed terminal X an element $v(X)$ of sort $type(X)$, i.e. of $\mathcal{B}$ or $\mathcal{R}$.

For each syntactic tree-structure τ of expressions $w \in L(A)$, $A \in N$, the *value* $[\![\tau]\!]_v^{\mathcal{P}}$ of τ in $\mathcal{P}$ under v is defined as follows:

1. If τ is the one-node tree of $w \in T$, then $[\![\tau]\!]_v^{\mathcal{P}} := v(w)$.
2. If τ is the combination of trees $\tau_{i_1}, \ldots, \tau_{i_n}$ by the rule $A \to A_1 \cdots A_k\ [t]$ with typed constituents $A_{i_1}, \ldots, A_{i_n}$, then
$$[\![\tau]\!]_v^{\mathcal{P}} := t^{\mathcal{P}}([\![\tau_{i_1}]\!]_v^{\mathcal{P}}, \ldots, [\![\tau_{i_n}]\!]_v^{\mathcal{P}}).$$

Typeless terminals in a rule r have syncategorematic meaning only, i.e. they are not assigned a value in $\mathcal{P}$, but are relevant for choosing t_r.

Example 37 *This is a fragment of the grammar of Böttner (1999).*

$$\begin{array}{rcccl} S & \to & UQ\ N\ VP\ [U(\overline{N} + VP)] & | & EQ\ N\ VP\ [E(N \cdot VP)] \\ VP & \to & IV\ [IV] & | & TVP\ [TVP] \\ TVP & \to & TV\ EQ\ N\ [TV : N] & | & TV\ UQ\ N\ [TV \dagger \overline{N}] \end{array}$$

The VP-rules say that the meaning of a VP is the meaning of the intransitive verb used, given by the evaluation $v : T \to \mathcal{P}$ of an interpretation, or the meaning of the transitive verb phrase. The first TVP-rule says that the meaning of a transitive verb followed by an existential quantifier and a common noun is the inverse image $[TV : N]$ of the set corresponding to N under the relation corresponding to TV. In the second S-rule, the quantifier EQ contributes to the sentence meaning by way of testing if the intersection of the N- and VP-meanings is nonempty. Note that EQ has different effects in the TVP- and S-rules.

Quantifiers and noun phrases, which in Montague grammar have higher-order functions as values, are not considered expressions in Peirce grammar; they have no meaning object attached at all. In fact, there is no way to have meaningful *NP*'s in Peirce grammar:

Example 38 *(Suppes (1976)) There is no category NP in Peirce grammar that subsumes quantified noun phrases EQ N and UQ N in the sense that*

$$\begin{array}{rcl} S & \to & UQ\ N\ VP\ [U(\overline{N} + VP)] \\ & | & EQ\ N\ VP\ [E(N \cdot VP)] \end{array} \tag{14.45}$$

would be equivalent (i.e. generate the same sentences with the same values) to the rules

$$\begin{array}{rcl} S & \to & NP\ VP\ [s(NP, VP)] \\ NP & \to & UQ\ N\ [u(N)] \quad | \quad EQ\ N\ [e(N)] \end{array} \tag{14.46}$$

for suitably chosen functions s, u, e on sets: in a Peirce algebra where $\mathcal{B}$ is the 2-element algebra $\mathbb{B}$, there are no functions s, u, e that give the expected values $s(u(0), 0) = U(\overline{0} + 0) = 1, \ldots, s(e(1), 1) = E(1 \cdot 1) = 1$.

Therefore, Peirce grammar prefers "flat" syntactic structures: since the value of an expression is computed from the first-order values of its immediate constituents, only flat structures provide enough arguments to the evaluation functions to make the necessary distinctions. Alternatively, one can use groupings with different category names:

Example 39 *Böttner (1999), rules 51–58, uses categories CPNP and APNP for phrases of conjunctions resp. alternatives of proper nouns, which get the same meaning but are treated differently when evaluating sentences:*

$$\begin{array}{rcl} CPNP & \to & PN\ \ Conj\ \ PN\ \ [PN + PN] \\ APNP & \to & PN\ \ Disj\ \ PN\ \ [PN + PN] \\ S & \to & CPNP\ \ VP\ \ [U(\overline{CPNP} + VP)] \\ & \mid & APNP\ \ VP\ \ [E(APNP \cdot VP)]\ . \end{array}$$

Thus, though "John and Mary" and "John or Mary" get the same set value, the two alternatives of the S-rule can use different Peirce-terms to define their meaning, as the syntactic construction of the subject is coded in its category name.

However, for nested occurrences of Conj and Disj, as in "John or Paul and Mary", collecting the constituents' meanings by set union $+$ would erase differences that must be maintained.

This trick of coding the structure of a phrase in its category name can only be used for finitely many different phrase structures; otherwise, we'd need infinitely many categories in the grammar.

14.3 Peirce grammar with trees and recursive evaluation

To overcome the above problems, we propose to allow finite trees as another sort of objects in Peirce algebras. More precisely, we define:

Definition 33 An *extended Peirce algebra* $\mathcal{P}' = (\mathcal{B}, \mathcal{R}, \mathcal{T}, :, ^c)$ is a Peirce algebra $\mathcal{P} = (\mathcal{B}, \mathcal{R}, :, ^c)$ expanded by the free algebra $\mathcal{T}$ of finite ordered trees generated from the elements of $\mathcal{B}$ and $\mathcal{R}$ by a finite set of constructor functions. An *extended Peirce term* is a term that is built from variables of the sorts *set*, *rel* and *tree* by means of the Peircean operations, the constructor functions, and operations that are defined by structural recursion on trees.

Definition 34 An *extended Peirce grammar* $PG = (G, type, e)$ is like a Peirce grammar, except that $type : N \cup T \to \{set, rel, tree\}$ may also assign the sort *tree* to terminals and nonterminals, and e assigns to each rule r of G an extended Peirce term of the appropriate type.

Example 40 *There is an extended Peirce grammar with rules (14.46) which is equivalent to the Peirce grammar with rules (14.45). In the rules of (14.46), let* $u, e : set \to tree$ *be unary constructor functions, and define* $s : tree \times set \to set$ *by recursion on its first argument:*

$$\begin{aligned} s(u(N), VP) &:= U(\overline{N} + VP), \ else, \\ s(e(N), VP) &:= E(N \cdot VP), \ else. \end{aligned}$$

These two clauses perform the case distinction on NPs which is impossible in Peirce algebra.

14.3.1 Coordination

In a higher-order setting such as the one of Montague (1974) or Keenan and Faltz (1985), the meaning of coordinated phrases can be defined pointwise, as is done by

$$[\![NP_1 \ and \ NP_2]\!] := \lambda P^{(e \to t)}([\![NP_1]\!](P) \wedge [\![NP_2]\!](P)),$$

for noun phrase meanings $[\![NP]\!] : (e \to t) \to t$, using the conjunction $\wedge$ on the type t of truth values. A drawback of this approach is that even if the basic types e of individuals and t of truth values have a decidable equality, the types of functions $(e \to t)$ and functionals $(e \to t) \to t$ do not. Moreover, the higher-order value is obtained by abstracting from contexts P of e-type positions in sentences, ignoring positions of different type, such as $e \times e$ for subjects of symmetric predicates, as in *John and Mary know each other*. Thus, a context-independent value for PN-conjunctions alone has sum-type $(e \to t) + (e \times e \to t)$ or worse, so the background type theory would at least have to be a bit more complicated than the one usually assumed.

In a first-order setting like Peirce algebra, one might use the boolean operations on the sets and relations to interprete coordinations. Böttner (1999), p.84, interprets *Conj* resp. *Disj* for categories *VP*, *TVP* (transitive verb phrase), *AP* and *PP* as intersection resp. union of sets or relations, but interprets both as union for proper and common nouns.

This does not work properly: (i) Conjunction of *PP*'s or *P*'s cannot be interpreted as intersection in *A tree was standing in front of and behind the house*, as the intersection of the two relations is empty. (ii) Conjunction of *N*'s cannot be interpreted as set union in *big (ants and elephants)*, where *big* has to be relativized to the two different nouns separately. (iii) Negation cannot be set complement resp. difference

in *John, but not Mary* or *men, but not women*, as then these would coincide with just *John* or *men*.

Since in contexts like *all (ants and elephants)*, conjunction of *N*'s *does* mean set union, while in (ii) it does not, the meaning of a coordinated expression may depend on the context of its use. Therefore, we better treat coordination as an abbreviation mechanism, reducing, in a context-dependent way, boolean combinations of phrases to boolean combinations of sentences.

When implementing this view by syntactic transformations, recursively embedded coordinations lead to recursive transformations, making it difficult to define the meaning of expressions by semantic attributes in the syntax rules (at least, of common tools like Yacc). Moreover, syntactic transformations must not copy expressions that lead to different values at different occurrences, or disturb quantifier scopes.

We propose a different route: don't expand coordinated phrases on the syntactic level, but during evaluation. For this it is necessary that from the semantic value of a coordinated phrase we can recover the form of the coordination and the meaning constituents. For example, to obtain

$$[\![(NP_1 \textit{ but not } NP_2)\ VP]\!] \;=\; [\![(NP_1\ VP) \textit{ and not } (NP_2\ VP)]\!],$$

from the values $[\![NP_1 \textit{ but not } NP_2]\!]$ and $[\![VP]\!]$ of the constituents on the left, we need to be able to read off the values of both constituent *NP*'s from that of the coordinated one, but also that the second one was negated.

Coordination rules. To treat coordination in extended Peirce grammar, we use binary constructor functions $\oplus$, $\odot$ and $\ominus$ to build trees of $\mathcal{T}$ with only *set*- or *rel*-values at their leaves. These constructors are analogs of the boolean operations $+$, $\cdot$ and $-$ (relative complement), used to delay evaluation.

The grammar rules may now use *set*- or *rel*-tree terms to define the value of coordinated expressions X' of category $X \in \{A, N^{pl}, IV, NP\}$:

$$\begin{aligned} X' \;&\to\; X \textit{ and } X\ [(X \odot X)], && (14.47)\\ &\mid\; X \textit{ or } X\ [(X \oplus X)], && (14.48)\\ &\mid\; X \textit{ but not } X\ [(X \ominus X)]. && (14.49) \end{aligned}$$

We think of the elements of $\mathcal{T}$ as second-class values. For example, while in $\mathcal{P}'$ nouns are considered first-class values belonging to $\mathcal{B}$ or $\mathcal{R}$, their coordinations are viewed as second-class values belonging to $\mathcal{T}$, such as $\textit{men} \ominus \textit{women}$ in *(men but not women) like football*, where the set difference $(\textit{men} - \textit{women}) \in \mathcal{B}$ clearly is wrong.

In a non-coordinating grammar rule, the evaluation function may

then recur along the tree structure of the values in $\mathcal{T}$ of its coordinated constituents. For example, in

$$S' \quad \rightarrow \quad NP'\ IV'\ [distr_{NP,IV}(NP', IV')] \tag{14.50}$$

we may "distribute" the IV' to the coordinates of the subject-NP and shift the connectives to the sentence level by clauses like

$$\begin{aligned} &distr_{NP,IV}((NP_1 \ominus NP_2), IV') \\ &\quad = \quad distr_{NP,IV}(NP_1, IV') \ominus distr_{NP,IV}(NP_2, IV'). \end{aligned}$$

When the value NP' in the subject is of the form $e(N)$ or $u(N)$ introduced by rules (14.46) and the value IV' is compound, the corresponding conjuncts can be interpreted by the boolean operations, as in

$$\begin{aligned} &distr_{NP,IV}(e(N), (IV_1 \odot IV_2)) \\ &\quad = \quad distr_{NP,IV}(e(N), (IV_1 \cdot IV_2)). \end{aligned}$$

When the value IV' is atomic, too, $distr_{NP,IV}$ coincides with the function s of example 40, so that, for example,

$$distr_{NP,IV}(e(N), IV) \quad = \quad E(N \cdot IV).$$

Note that $distr_{NP,IV}$ operates as a homomorphism on $\mathcal{T}$ in its first argument. By using rule-specific functions *distr*, the constructors $\odot, \oplus, \ominus$ may influence the meaning of expressions in a context-dependent way.

Coordination of nouns and adjectives. When several constituents in the same rule are coordinations, there is a question of relative scope of the connectives. If a constituent is to get narrow scope, the evaluation function has to recur on this constituent later. For example, to give the combinations of N's higher precedence over those of classficatory adjectives A's in modified nouns, for the rule

$$N' \quad \rightarrow \quad A'\ N'\ [distr_{A,N}(A', N')] \tag{14.51}$$

we define $distr^{pl}_{A,N} : tree \times tree \rightarrow tree$ for $\circ \in \{\odot, \oplus, \ominus\}$ as follows[1]:

$$\begin{aligned} distr^{pl}_{A,N}((A_1 \circ A_2), N) \quad &= \quad distr^{pl}_{A,N}(A_1, N) \circ distr^{pl}_{A,N}(A_2, N)), \\ distr^{pl}_{A,N}(A, (N_1 \circ N_2)) \quad &= \quad distr^{pl}_{A,N}(A, N_1) \circ distr^{pl}_{A,N}(A, N_2)), \\ distr^{pl}_{A,N}(A, N) \quad &= \quad A \cdot N, \qquad \text{else.} \end{aligned} \tag{14.52}$$

[1]We restrict these to plural, since the corresponding clauses of $distr^{sg}_{A,N}$ for singular might differ for *and* at least.

Then every interpretation of this grammar satisfies the equation

$$\begin{aligned}
&[\![\textit{(white but not grey) (ants and elephants)}]\!] \\
&\quad = \quad \textit{distr}^{pl}_{A,N}(\textit{white} \ominus \textit{grey}, \textit{ant} \odot \textit{elephant}) \\
&\quad = \quad \textit{distr}^{pl}_{A,N}(\textit{white}, \textit{ant} \odot \textit{elephant}) \ominus \textit{distr}^{pl}_{A,N}(\textit{grey}, \textit{ant} \odot \textit{elephant}) \\
&\quad = \quad ((\textit{white} \cdot \textit{ant}) \odot (\textit{white} \cdot \textit{elephant})) \ominus ((\textit{grey} \cdot \textit{ant}) \odot \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\textit{grey} \cdot \textit{elephant})) \\
&\quad = \quad [\![\textit{(white (ants and elephants)) but not} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textit{(grey (ants and elephants))}]\!]
\end{aligned}$$

Note, by the way, that this equality does not depend on expansions of coordinated N's in sentences to sentence coordinations.

Ordinary Peirce grammar (cf. Böttner (1999), rule 57) interprets conjunction of nouns by set union $+$, so that, by laws of boolean algebra,

$$\begin{aligned}
[\![\textit{white (ants and elephants)}]\!] \quad &= \quad \textit{white} \cdot (\textit{ant} + \textit{elephant}) \\
&= \quad (\textit{white} \cdot \textit{ant}) + (\textit{white} \cdot \textit{elephant}).
\end{aligned}$$

This seems simpler than going via the above clause of $\textit{distr}_{A,N}$ for classificatory adjectives, i.e.

$$\textit{distr}^{pl}_{A,N}(\textit{white}, \textit{ant} \odot \textit{elephant}) = (\textit{white} \cdot \textit{ant}) \odot (\textit{white} \cdot \textit{elephant}),$$

which keeps the component sets $(\textit{white} \cdot \textit{ant})$ and $(\textit{white} \cdot \textit{elphant})$ apart.

The reason for already keeping the components in $\textit{ant} \odot \textit{elephant}$ apart is that the conjunction of common nouns is *not* set union in the scope of intensive adjectives (IA), which denote partial order relations: in *big (ants and elephants)*, we have to relativize –by a suitable operation r– the relation *big* to the sets *ant* and *elephant* separately and then take the union of the result, so that

$$\begin{aligned}
[\![\textit{big (ants and elephants)}]\!] \quad &= \quad r(\textit{big}, \textit{ant}) + r(\textit{big}, \textit{elephant}) \\
&\neq \quad r(\textit{big}, \textit{ant} + \textit{elephant}),
\end{aligned}$$

where only the set on the left contains some ants. In extended Peirce grammar, we can model this by a rule

$$N' \quad \rightarrow \quad IA'\ N'\ [\textit{distr}_{IA,N}(IA', N')] \tag{14.53}$$

with $\textit{distr}_{IA,N}$ differing from $\textit{distr}_{A,N}$ above by replacing (14.52) with

$$\textit{distr}^{pl}_{IA,N}(IA, N) \quad := \quad r(IA, N),$$

where the relativization $r(R, B) := h(R \cdot (B \times B)) : 1$ is the domain of the "left half" h of the restriction of R to B^2. So, as with classificatory

[2] By the "left half" $h(S)$ of a binary relation S we mean its edges from minimal

adjectives, we distribute the intensive adjective to conjuncts by

$$distr^{pl}_{IA,N}(big, ant \odot elephant) = r(big, ant) \odot r(big, elephant),$$

but do not and must not interprete noun conjunction as set union in the scope of an intensive adjective, as

$$r(big, ant + elephant) \neq r(big, ant) + r(big, elephant).$$

The relativization r operates in its second argument as a homomorphism on $\mathcal{T}$, but not on the first-class values $\mathcal{B}$. Note also that r allows us to interprete the attributive use of intensive adjectives while avoiding the comparison with artificial elements like "the medium-sized elephant" of Böttner (1999). Depending only on the values of nouns and the global *bigger-than*-relation with its converse *smaller-than*, we may obtain *big (white elephants)* $\neq$ *white (big elephants)*, *big ants* $\subseteq$ *small animals*, *small (big elephants)* $\subset$ *big elephants* etc.

Thus, noun conjunction is treated differently in the scope of classificatory and intensive adjectives; the pseudo-values of *set*-trees and *rel*-trees can be used to define the meaning of expressions containing coordinations by taking the context of the coordination into account.

14.3.2 Inverse linking

The question of relative scope for coordination connectives mentioned above arises also for coordinations of verbs applied to coordinated noun phrase arguments, as in *John or Mary could hear, but not see a man and a woman shout.* To give the subject *NP* widest scope, in

$$S \to NP\ \ VP\ [distr_{NP,VP}(NP, VP)]$$

we let $distr_{NP,VP}$ recur on the *NP*-structure first, and in the *VP*-rule let the object *NP* have scope over the predicate (see (14.54),(14.55) below). Then $distr_{NP,VP}$ also gives quantifiers in the subject wide scope over those in the object, subsuming the special cases of example 37.

(According to this scheme, active and passive of simple sentences with transitive verb and coordinated or quantified *NP*-complements

---

nodes, $S \cap (\overline{ran(S)} \times ran(S))$, and the left half of S without edges from minimal or to maximal nodes, $h(S \cap (ran(S) \times dom(S)))$. More precisely, we define h by

$$h(S) := \begin{cases} \emptyset, & \text{if } S = S \cdot (\breve{S} : 1 \times S : 1) \\ S \cdot (\overline{\breve{S} : 1} \times \breve{S} : 1) \ + \ h(S \cdot (\breve{S} : 1 \times S : 1)), & \text{else.} \end{cases}$$

In particular, if $\leq$ is a finite linear order, then $h(\leq)$ consists of the $\leq$-edges leaving the smaller half of the underlying set. Since h recurs along the inclusion $\subseteq$ of relations and not along the structure of trees, we need to relax our definitions a bit to turn $h(S)$ and hence $r(R, B)$ into an extended Peirce term. Note that $r(R, B_1 + B_2)$ is not the sum or any other algebraic function of $r(R, B_1)$ and $r(R, B_2)$.

get different meanings, e.g. in *every child loves some adults* versus *some adults are loved by every child.*)

To handle scope relations that cannot be read off from the constituent structure, we have to relax the uniqueness of the term assignment $e(r) = t_r$ and admit different t_r implementing different scope readings for the same grammar rule r. A case in question is the scope relation between dative and accusative object of a ditransitive verb; this is treated in Böttner (1999), rules R 105-115, by an extension of ordinary Peirce algebra to ternary relations. Here we consider a more difficult case of quantifier scope in nested *NP*s.

Peirce's operations model combinations of relational and common nouns like

$$\begin{aligned} \textit{brother of a student} &= B : S \\ \textit{friend of every brother of a student} &= \overline{\overline{F} : (B : S)} = F \dagger \overline{(B : S)}. \end{aligned}$$

These amount to a reading where the syntactically innermost quantifier has narrowest scope:

$$\begin{aligned} &\textit{(Some reader of (every paper of (some author))) got bored} \\ &=_{a)} \quad \exists x_1(\forall x_2(\exists x_3(A(x_3) \wedge P(x_2, x_3)) \rightarrow R(x_1, x_2)) \wedge B(x_1)). \end{aligned}$$

However, there is a different reading of *NP*'s of this form, where the syntactically innermost quantifier gets widest scope, the so-called *inverse linking* construction:

$$\begin{aligned} &\textit{(Some reader of (every paper of (some author))) got bored} \\ &=_{b)} \quad \exists x_3(A(x_3) \wedge \forall x_2(P(x_2, x_3) \rightarrow \exists x_1(R(x_1, x_2) \wedge B(x_1)))). \end{aligned}$$

The inversion of quantifiers shows nicely when this is written with bounded quantifiers as

$$\begin{aligned} &(Q_1\ RN_1\ \textit{of}\ (Q_2\ RN_2\ \textit{of}(\ldots \textit{of}\ (Q_n N)\ldots)))\ IV \\ &\quad = \quad (Q_n x_n \in N)\ldots(Q_2 x_2 RN_2 x_3)(Q_1 x_n RN_1 x_2)\ IV(x_n). \end{aligned}$$

In his review of Böttner (1999), Jäger (2004) notes that Peirce grammar seems unable to treat the inverse linking construction and could do so at best with infinitely many categories or grammar rules.

Let us see how this construction can be handled in extended Peirce grammar. To highlight the role of the quantifiers Q in the algebraic combinations, let $\overset{Q}{c}$ (with c for copula) be the operation between two sets with Q in the subject:

$$A \overset{\forall}{c} B \ := \ U(\overline{A} + B), \qquad A \overset{\exists}{c} B \ := \ E(A \cdot B).$$

Likewise, let $\overset{Q}{?}$ be the module operation between a relation and a set

with Q in the object:

$$R \overset{\exists}{\cdot} A \ := R : A, \qquad R \overset{\forall}{\cdot} A \ = R \dot{+} \overline{A}.$$

By means of these, we can translate the linking construction of *NP*'s in subject position as in:

$$\begin{aligned}
(Q\ N)\ IV &\mapsto N \overset{Q}{c} IV \\
(Q_1\ RN_1\ \mathit{of}\ (Q_2\ N))\ IV &\mapsto N \overset{Q_2}{c} (RN_1^{\smile} \overset{Q_1}{\cdot} IV) \\
(Q_1\ RN_1\ \mathit{of}\ (Q_2\ RN_2\ \mathit{of}\ (Q_3\ N)))\ IV &\mapsto N \overset{Q_3}{c} (RN_2^{\smile} \overset{Q_2}{\cdot} (RN_1^{\smile} \overset{Q_1}{\cdot} IV)).
\end{aligned}$$

Thus, if the value of the subject *NP* in a sentence *NP VP* codes the syntactic structure of the *NP* together with the meaning of the nouns used, the modified Peirce grammar rule is

$$S \quad \rightarrow \quad NP\ VP\ [\mathit{distr}_{NP,VP}(NP, VP)] \tag{14.54}$$

where the function $\mathit{distr}_{NP,VP}$ (extending the s of above) recurs along the *NP*-structure by:

$$\begin{aligned}
\mathit{distr}_{NP,VP}((Q\ N), VP) &:= N \overset{Q}{c} VP, \\
\mathit{distr}_{NP,VP}((Q\ RN\ \mathit{of}\ NP), VP) &:= \mathit{distr}_{NP,VP}(NP, RN^{\smile} \overset{Q}{\cdot} VP).
\end{aligned}$$

Note that we need the converse of the relational noun. For example,

$$\begin{aligned}
&\textit{(Some reader of (every paper)) gets bored} \\
&= (\exists\ R\ \mathit{of}\ (\forall\ P))\ B \\
&= \mathit{distr}_{NP,VP}((\exists\ R\ \mathit{of}\ (\forall\ P)), B) \\
&= \mathit{distr}_{NP,VP}((\forall\ P), \breve{R} \overset{\exists}{\cdot} B) \\
&= P \overset{\forall}{c} (\breve{R} \overset{\exists}{\cdot} B) \\
&= \forall y(P(y) \rightarrow \exists x(\breve{R}(y,x) \wedge B(x))) \\
&= \forall y(P(y) \rightarrow \exists x(R(x,y) \wedge B(x)))
\end{aligned}$$

To handle the inverse linking construction in *NP*s in object position, we need relative operations $\overset{Q}{;}$ determined by the quantifiers Q, combining two relations:

$$R \overset{\exists}{;} S := R ; S, \qquad R \overset{\forall}{;} S := R \dot{+} \overline{S}.$$

A different translation is needed, which may return a set $\neq 0, 1$:

$$\begin{aligned}
TV\,(Q\ N) &\mapsto TV \overset{Q}{\cdot} N \\
TV\,(Q_1\ RN_1\ (Q_2\ N_2)) &\mapsto (TV \overset{Q_1}{;} RN_1)\ (Q_2\ N_2) \\
TV\,(Q_1\ RN_1\ (Q_2\ RN_2\ (Q_3\ N))) &\mapsto ((TV \overset{Q_1}{;} RN_1) \overset{Q_2}{;} RN_2) \overset{Q_3}{\cdot} N_3
\end{aligned}$$

Again, the innermost quantifier has widest scope. The grammar rule is

$$TVP \ \rightarrow \ TV\ NP\ [\mathit{distr}_{TV,NP}(TV, NP)] \tag{14.55}$$

where $distr_{TV,NP}$ recurs along the structure of the object-NP by

$$\begin{aligned} distr_{TV,NP}(TV, (Q\ N)) &:= TV \overset{Q}{?} N, \\ distr_{TV,NP}(TV, (Q\ RN\ of\ NP)) &:= distr_{TV,NP}(TV \overset{Q}{?} RN,\ NP). \end{aligned}$$

For example, this gives

$$\begin{aligned} &x\ likes\ (every\ paper\ of\ (some\ author)) = x\ L\ (\forall\ P\ of\ (\exists\ A)) \\ &= ((L \overset{\forall}{?} P) \overset{\exists}{?} A)\ (x) = \exists z(\forall y(L(x,y) \leftarrow P(y,z)) \wedge A(z)). \end{aligned}$$

Finally, note that for arguments $(Q\ RN\ of\ NP)$ where the NP is a conjunction, e.g. *a friend of John and Mary*, inverting the scopes by

$$distr_{NP,VP}((Q\ RN\ of\ NP), VP) = distr_{NP,VP}(NP,\ RN^{\smile} \overset{Q}{?} VP)$$

produces the wrong reading with possibly different friends. The analysis where first the NP is combined with RN by

$$N' \rightarrow RN\ of\ NP\ [distr_{RN,NP}(RN, NP)] \tag{14.56}$$

gives the reading with a common friend when $distr_{RN,NP}$ recurs along its second argument with clauses like

$$\begin{aligned} distr_{RN,NP}(RN, (PN_1 \odot PN_2)) &= RN \overset{\forall}{?} (PN_1 + PN_2), \\ distr_{RN,NP}(RN, (PN_1 \oplus PN_2)) &= RN \overset{\exists}{?} (PN_1 + PN_2). \end{aligned}$$

Again, the coordination constructor influences the evaluation "later".

14.4 Comparsion with other approaches

Peirce grammar differs from other grammatical theories in that meanings are first-order objects, abstract sets and relations, which can only be composed by algebraic operations. Extended Peirce grammar adds a further sort of meanings, finite trees of sets and relations, from which constituent meanings can be extracted. These "second-class" values have no ontological motivation — they only serve as intermediate stages in the evaluation of sentences, allowing us to give the context of an expression an access to the meaning *constituents* of the expression.

This makes a major difference both to the relational semantics of ordinary Peirce grammar and the higher-order semantics of Montague or categorial grammar: neither the boolean and peircean operations can be reversed, nor function application and abstraction. Thus, when a noun coordination like *ants and elephants* is interpreted by set union $ant + elephant$ or pointwise function application $\lambda x(ant(x) \vee elephant(x))$, we cannot get back the meaning constituents *ant* and *elephant*. In extended Peirce grammar, we can get them back from the second-class value $ant \odot elephant$ and hence we can correctly compute the relativization *big (ants and elephants)*.

Although the Boolean semantics of Keenan and Faltz (1985) is also higher-order, the difference to extended Peirce grammar is not that big. In Boolean semantics, every category C of conjoinable phrases comes with a boolean algebra $\mathcal{B}_C$ of possible values, and the connectives *and*, *or*, and *not* for C-phrases are interpreted by the operations of $\mathcal{B}_C$. The point is that for categories of function type D/C, say, the algebra $\mathcal{B}_{D/C}$ does not contain arbitrary functions from $\mathcal{B}_C$ to $\mathcal{B}_D$, but boolean homomorphisms only; hence, each $f \in \mathcal{B}_{D/C}$ has access to the meaning constituents of its argument, as in $f(c_1 + c_2) = f(c_1) + f(c_2)$.

While a Peirce algebra only has the two boolean algebras $\mathcal{B}$ of sets and $\mathcal{R}$ of binary relations, the tree algebra $\mathcal{T}$ of an extended Peirce algebra can be chosen to subsume term algebras $\mathcal{T}_C$ of phrases of category C. The evaluation terms in our grammar rules have mostly been used to define homomorphisms between such term algebras $\mathcal{T}_C$. We expect that a technical connection can be established between the quotients of $\mathcal{T}_C$ modulo the boolean algebra axioms and the boolean algebras $\mathcal{B}_C$ of Keenan and Faltz, provided we impose some invariance restrictions on the evaluation terms. But, unlike Boolean semantics, extended Peirce grammar does not assume that all categories of conjoinable expressions are interpreted in an ontologically well-motivated boolean algebra.

There is a minor difference to Boolean semantics concerning the implicit scope rules of natural language. Consider simple sentences built by applying a boolean combination (BC) of transitive verbs to boolean combinations of noun phrases,

$$BC(TVs)(BC(NPs), BC(NPs)).$$

We expect this to be evaluated by a boolean algebra homomorphism in each coordinate, i.e. for *atomic* expressions v, e, we have

$$v(e, BC(c_1, \ldots, c_r)) = BC(v(e, c_1), \ldots, v(e, c_r))$$

with the same boolean combination, and likewise for subject and predicate. What remains not completely determined, e.g. in German, is the relative scope of the quantifiers and connectives in the predicate and argument positions. Keenan and Faltz (1985) built specific scope rules into their interpretation, which we also did in the examples, but in general there are different readings which may be explicitly marked, as in *John and Mary are working on a (common) paper* vs. *John and Mary are working on a paper (each)*. In general, a non-deterministic evaluation is needed that chooses between various possible scope relations.

The simple algebraic semantics of Peirce grammar clearly poses severe restrictions, viz. not all first-order formulas built from unary and binary relations are expressible as Peircean terms. Extensions to rela-

tions of arity ≥ 3 are possible, but an equational axiomatization with additional primitives, like argument permutation, seems missing. In Peirce grammar, one can define the *at least two*-quantifier $\exists^{\geq 2}$ via the diversity relation $\overline{I}$ and thus handle a distributive reading of plurals. But it is impossible to define *at least* n for $n \geq 3$, unless, for example, one admits n-ary relations with a constant for n-ary diversity.

What about generalized quantifiers like *most*? In the algebra $\mathcal{P}_V$ of all subsets and relations on V one can define the copula-, module- and relation-operations $\overset{M}{c}$, $\overset{M}{\cdot}$, $\overset{M}{,}$ from the set-theoretic definition $M(A,B) :\Longleftrightarrow\ |A \cdot B| > |A \cdot \overline{B}|$ of the quantifier M, and then evaluate simple sentences of the form $(Q_1\ N_1)\ TV\ (Q_2\ N_2)$ as before by $N_1 \overset{Q_1}{c} (TV \overset{Q_2}{\cdot} N_2)$. But an algebraic treatment of *most* would need an equational axiomatization of $\overset{M}{c}$, $\overset{M}{\cdot}$, $\overset{M}{,}$, which does not seem to exist.

Up to now, Peirce grammar has, to my knowledge, only been used to model extensional features of natural language. In particular, it is not clear how to handle verbs with propositional arguments.

Concerning syntax, the proposed extension of Peirce grammar allows us to handle recursive syntactic constructions and extensively use traditional constituent structures of context-free grammars. This is arguably a step away from Peirce's original emphasis on n-ary relations and their combinations, which is closer in spirit to dependency grammar.

We have only dealt with coordination of words or constituents. In transformational grammar, "non-constituent"-coordinations have been considered the result of transformations of coordinations, such as right or left peripheral extraction or argument cluster coordination:

(RPE) $(\alpha\gamma)\ Conj\ (\beta\gamma)\ \rightsquigarrow\ [\alpha\ Conj\ \beta]\,\gamma$

(LPE) $(\gamma\alpha)\ Conj\ (\gamma\beta)\ \rightsquigarrow\ \gamma\ [\alpha\ Conj\ \beta]$

(ACC) $(\alpha\gamma\beta)\ Conj\ (\alpha'\gamma\beta')\ \rightsquigarrow\ \alpha\gamma\beta\ Conj\ [\alpha'\beta']$,

where restrictions on the categorial status of the expression sequences $\alpha, \beta, \gamma, \alpha', \beta'$ have been the subject of debate. At least simple cases of these can be handled in Peirce grammar using "flat" syntax rules. For example, the reading of

(RPE) *[all men like, but some women dislike] some jokes*

where *some jokes* has wide scope and cannot be distributed "back", gets its value by the Peirce term $E(J \cdot ((L^{\smile} \overset{\forall}{\cdot} M) \cdot (\overline{L}^{\smile} \overset{\exists}{\cdot} W)))$, but I did not consider more complicated NPs here. For examples like

(ACC) *Mary wrote a paper and [(John and Bill) (two reviews)]*,

constructed by a "flat" syntax-rule

$$S \ \rightarrow\ NP_1^{nom}\ TV\ NP_2^{acc}\ Conj\ NP_3^{nom}\ NP_4^{acc}\ [t],$$

the value can be defined by the extended Peirce-term

$$t := f(NP_1, TV, NP_2) \wedge f(NP_3, TV, NP_4)$$

where f is one of the following, depending on whether coordinations and quantifiers in the subject are to have wide scope or not:

$$\begin{aligned} f(NP_1, TV, NP_2) &:= distr_{NP,VP}(NP_1, distr_{TV,NP}(TV, NP_2)), \\ f(NP_1, TV, NP_2) &:= distr_{NP,VP}(NP_2, distr_{TV,NP}(TV^{\smile}, NP_1)). \end{aligned}$$

Note that the value TV from the initial constituent $(NP_1\ TV\ NP_2)$ is reused to evaluate the non-constituent $[NP_3\ NP_4]$.

This sketchy treatment of non-constituent coordination seems more in line with ideas used in HPSG, see Beavers and Sag (2004), than with those of combinatory categorial grammar, which rely on function types. A treatment with product types is proposed in Houtman (1994).

14.5 Conclusion and open problems

We have proposed a solution to what was called the 'one big thread to the entire project of relational grammar' in Jäger (2004), recursivity. Essentially, we turn syntactic structure to values and admit evaluation of expressions by structural recursion on those values, keeping the grammar finite. For programming languages, where programs are large, this would give values of intolerable size. For natural language, where sentences are short, the tree values are shallow. However, if their leaves are sets and relations in extension, we may also have a size and hence complexity problem. So we better interprete the grammar in a Peirce algebra $\mathcal{P}$ where sets and relations are abstract elements, not in a full algebra $\mathcal{P}_V$. It deserves to be investigated if we can turn the definable elements of a given $\mathcal{P}_V$ into an abstract interpretation $\mathcal{P}$ with efficiently computable operations.

The examples given show that our proposal is a useful and necessary modification of Peirce grammar, but it needs to be clarified what we have to pay. By adding the sort of trees we loose the equational axiomatizability of Peirce algebra, since inequational Horn formulas are needed to state the freeness assumptions for constructors (c.f. Maher (1988)). Using the semantic annotations of grammar rules, we may check the semantic validity of inferences based on grammatical form. While this can be done by equational reasoning for Peirce grammar, for extended Peirce grammar we will also need to reason by induction on the tree structure, which seems acceptable. A less tolerable step away from the equational theory of Peirce algebra is our use of recursion on the subsumption order to define the relativization operation.

Our extension may be (mis)used to deviate radically from the strictly

bottom-up way of evaluation in ordinary Peirce grammar: one can delay evaluation to first-class values until the complete constituent structure of an expression is mirrored in an intermediate second-class value.

Finally, trees as values offer an advantage over higher-order semantic values of other frameworks: one may look for decidable relations between trees that imply semantic equivalence. This also can be relevant for checking the validity of grammatical inferences.

Acknowledgement

I wish to thank the referees for critical questions and Michael Böttner for copies of papers on Peirce and for keeping in touch over the years.

References

Beavers, John and Ivan A. Sag. 2004. Coordinate ellipsis and apparent non-constituent coordination. In S. Müller, ed., *Proceedings of the HPSG-04 Conference*. Stanford University, CSLI Publications.

Böttner, Michael. 1999. *Relationale Grammatik*. Tübingen: Max Niemeyer Verlag.

Brink, Chris, Katharina Britz, and Renate A. Schmidt. 1994. Peirce algebras. *Formal Aspects of Computing* 6:339–358.

Houtman, Joop. 1994. *Coordination and Constituency. A Study in Categorial Grammar*. Ph.D. thesis, Rijksuniversiteit Groningen.

Jäger, Gerhard. 2004. Book Review of "Relationale Grammatik" by Michael Böttner. *Journal of Logic, Language and Information* 13(4):521–525.

Keenan, Edward L. and Leonard M. Faltz. 1985. *Boolean Semantics for Natural Language*. Dordrecht: D. Reidel.

Maher, M. J. 1988. Complete axiomatizations of the algebras of finite, rational and infinite trees. In *3rd IEEE Symposium on Logic in Computer Science*, pages 348–357. IEEE Computer Society Press.

Montague, Richard. 1974. The proper treatment of quantification in ordinary english. In R. Thomason, ed., *Formal Philosophy. Selected Papers of Richard Montague*. Yale Univ. Press.

Peirce, Charles Sanders. 1932. *Collected Papers*, vol. III, chap. The Logic of Relatives, pages 228–345. Cambridge, Mass.: Harvard University Press. (Originally published in: The Monist, vol.7, pp.161–217, 1897).

Suppes, Patrick. 1976. Elimination of quantifiers in the semantics of natural language by use of extended relation algebras. *Revue Internationale de Philosophie* 117/118:243–259.