Nominal Compounds, the Principle of Compositionality and Conceptual Graphs

Karl Heinz Wagner
Universität Bremen

1 Introduction

The starting point for the following remarks is WOLFGANG WILDGEN'S paper "Zur Dynamik lokaler Kompositionsprozesse: Am Beispiel nominaler *ad hoc*-Komposita" (WILDGEN 1982) written more than twenty years ago. Wildgen begins his paper with the following statement:

> Nominal composition still is a challenge to the available models of grammar, since on the one hand it exhibits quasi-syntactic properties and thus is comparable to noun phrases and relative clause constructions, yet on the other hand the result of the composition process is a word and shares the phonological and semantic integrity of words. Particularly problematic is the description of *ad-hoc* compounds, which cannot be assigned a marginal place in the grammar as mere lexical phenomena (free translation: KHW).[1]

Wildgen then formulates a number of assumptions ("Thesen") concerning the composition process, the first three of which will play a role in the following discussion:

> *Assumption 1:* The distinction between relational and substantival constituents [...] is an essential basis for the semantic aspect of the composition process.[2]
>
> *Assumption 2:* The functional incompleteness of non-relational compounds creates a semantic wake whose result is that
>
> (a) the information encapsulated in the word form is exploited,
>
> (b) co-textual and contextual information is increasingly spliced into the composition process.
>
> If both sources become unavailable the speaker/hearer can enrich the lexical semantics and the context by his own invention.[3]
>
> *Assumption 3:* The semantics of the constituents of the compound which is made accessible by the functional wake has several levels of accessibility. The relations or predicate constants derived from sentences or propositions in traditional research on compounds can be obtained from the lexical semantics (unless they are explicated by co-texts and contexts).[4]

In contrast to lexicalized compounds, whose idiosyncratic properties render their semantics increasingly opaque and non-compositional, *ad-hoc* compounds in order for them to be interpretable by the hearer must obey some version of the principle of compositionality. This

[1] Die nominale Komposition stellt immer noch eine Herausforderung an die vorhandenen Grammatikmodelle dar, da sie einerseits quasi-syntaktische Eigenschaften aufweist und insofern mit Nominalphrasen und Relativsatzkonstruktionen vergleichbar ist, andererseits ist das Ergebnis des Kompositionsprozesses ein Wort und es teilt die phonologische und semantische Ganzheitlichkeit von Wörtern. Besonders problematisch ist die Beschreibung von *ad hoc*-Komposita, die man nicht als lexikalische Phänomene in Randbereiche der Grammatik abschieben kann (WILDGEN: 297).

[2] These 1: Die Unterscheidung zwischen relationalen und substantivischen Konstituenten [...] ist eine wesentliche Grundlage für den semantischen Aspekt des Kompositionsprozesses (WILDGEN: 298).

3 These 2: Die funktionale Unvollständigkeit nichtrelationaler Komposita erzeugt einen semantischen Sog, welcher dazu führt, daß
(a) die in der Wortgestalt verkapselte Information ausgebeutet wird,
(b) ko- und kontextuelle Informationen verstärkt in den Kompositionsprozeß einfließen.
Wenn beide Quellen versiegen, kann der Sprecher/Hörer noch die Wortsemantik und den Kontext durch freie Erfindung anreichern (WILDGEN: 300).

[4] These 3: Die Semantik der Konstituenten des Kompositums, welche durch den funktionalen Sog erschlossen wird, hat verschiedene Zugänglichkeitsebenen. Die in der traditionellen Kompositionsforschung aus Sätzen oder Satzbegriffen abgeleiteten Relationen bzw. Prädikatskonstanten können aus der Wortsemantik gewonnen werden (sofern sie nicht durch Ko- und Kontexte expliziert sind)(WILDGEN: 301).

principle assumes that meaning of an expression is derivable from the meanings of its constituent elements together with its syntactic structure and surrounding context.

This is not the place to review the many proposals that have been made to account for the various problems one encounters when dealing with nominal compounds. These proposals include frameworks as diverse as *case grammar* (HÜLLEN 1976:71–95;WAGNER 1971; KÜRSCHNER 1974), *Montague grammar* (FANSELOW 1981), *Discourse Representation Theory (DRT)* (MEYER 1993), or even *Head Driven Phrase Structure Grammar (HPSG)* (REINHARD 2001).

In the following it will be argued that the theory of CONCEPTUAL GRAPHS (CGs) which has been developed by John F. Sowa since the 1970s and has gained growing influence particularly after the publication of his first book (SOWA 1984; cf. also SOWA 1988; 1991; 1992; 1993a; 1993b; and 2000) is sufficiently rich to account for many of the problems discussed in WILDGEN'S paper in a comprehensive and unified manner, and includes many of the advantages of other frameworks, notably *case grammar*, *Montague grammar*, and *DRT*. For example, the distinction mentioned above between *relational* and *substantival* constituents and their role in the interpretation of *ad-hoc* compounds can be accounted for by *role types* and *natural types*, respectively. The theory provides precise representations and operations defined on them that enable us to make the interaction between lexical semantics and background knowledge explicit and transparent.

2 Conceptual Graphs

CONCEPTUAL GRAPHS (CGs) are a system of logic which is based on the EXISTENTIAL GRAPHS of CHARLES SANDERS PEIRCE[5], the semantic networks of artificial intelligence and cognitive psychology, and contributions from linguistics. What makes them attractive is that they express meaning in a form that is logically precise, computationally tractable, without sacrificing human readability. Conceptual graphs have been implemented in a number of computational projects including natural language processing.

In a nutshell, the system of conceptual graphs makes use of the following formal objects (cf. SOWA & WAY 1986: 59; more details in SOWA 1992):

- *Concepts* with type labels and referents.
- *Conceptual relations* with type labels and arcs.
- *Conceptual graphs*, which consist of concepts linked by relations.
- *Contexts*, which are complex concepts of type proposition or situation and permit the nesting of conceptual graphs to express negation, modality, tense, and propositional attitudes (belief, certainty, uncertainty, etc.).
- *Lambda abstractions*, which are parameterized conceptual graphs used in the definition of *types, schemata,* or *prototypes*.
- A *lexicon* for associating word forms with their syntactic categories and concept types.

Other objects in the theory such as *canonical graphs*, which will play an important role in the discussion below, as well as *type definitions*, are derived from these in a controlled way. Together they form a rich SEMANTIC NETWORK of general information about some domain of discourse, which can be drawn upon in the interpretation of nominal compounds.

[5] There is an accessible tutorial text by Peirce himself (PEIRCE 1909) with a commentary by John F. Sowa. For a more detailed discussion see ROBERTS (1973) and SHIN (2002).

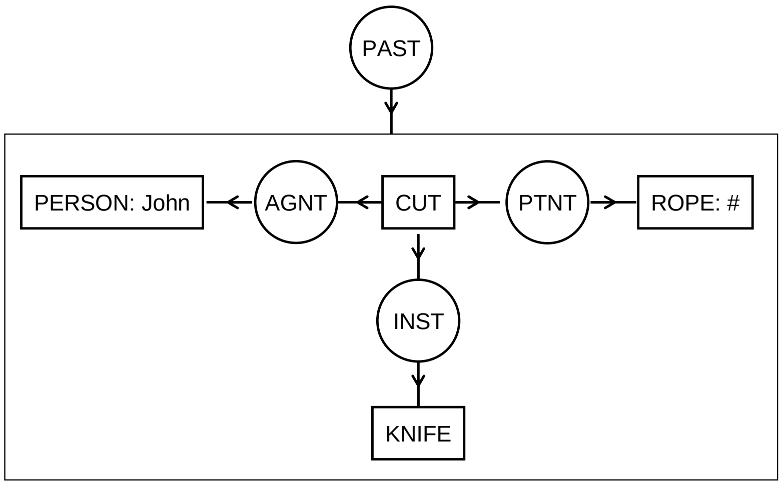

Figure 1. A conceptual graph in display form

A conceptual graph is an abstract structure that can be used to represent the meaning of an expression such as a sentence. Conceptual graphs are formally defined by an abstract syntax that is independent of any notation and can be visualized in different ways.[6] The most transparent realisations are diagrams like Figure 1 (*display form* DF) representing the sentence *John cut the rope with a knife*. Besides this graphical form there is also a *linear form* (LF), which is intended as a more compact notation than DF, retaining good human readability. It is exactly equivalent in expressive power to the abstract syntax and the display form. Figure 2 shows the LF for Figure 1.

```
(PAST) → [[CUT] –
                  (AGNT) → [PERSON: John]
                  (PTNT) → [ROPE: #]
                  (INST) → [KNIFE]].
```

Figure 2. A conceptual graph in linear form

In graph-theoretical terms a conceptual graph is a finite connected directed bipartite graph. A graph is called bipartite if the nodes it contains can be partitioned into two disjoint sets. In the case of conceptual graphs these are concept nodes (represented by boxes in DF or square brackets in LF) and relation nodes (represented by circles in DF or round parentheses in LF).[7] The conceptual graph in Figure 1 contains four concepts: [PERSON: John] and [ROPE: #] refer to specific instances of a person and a rope, whereas [CUT] refers to some unspecified instance of cutting; and [KNIFE] refers to an unspecified knife.[8]

2.1 Concepts

Concepts represent the meanings of lexemes. They may refer to entities, properties, events, activities or actions in the world. Every concept has a *type* and a *referent*. To distinguish the *type* of the referent from the specific individual, the concept box is divided into two parts: a *type field* and a *referent field*: [<type>:<referent>]. For example, the concept [PERSON: John] is an *individual concept* with *type* PERSON and *referent* John. The concepts [KNIFE] and [CUT] are called *generic concepts*, because they specify only the type but do not identify a particular individual.

[6] A definition of the abstract syntax can be found in an ISO-Standard proposal for CGs edited by John F. Sowa (cf. URL: http://www.jfsowa.com/cg/cgstand.htm; see also Sowa 2000:476–491).

[7] The exact shapes are irrelevant. "Circles" often take the form of ellipses to accommodate longer labels.

[8] Since conceptual graphs are a system of logic, they can be translated to other versions of logic, such as predicate calculus (cf. SOWA 1992).

2.2 Referents

In the *basic notation* for conceptual graphs only three kinds of referents are available:

- *Existential referents*, represented by the symbol *, indicating that there exists at least one individual of the appropriate type. It thus corresponds to the existential quantifier ∃ in formal logic. The full notation [KNIFE: *] can be abbreviated to [KNIFE].
- *Individual markers,* represented by the symbol # followed by a positive integer, e.g. #32415 (identification number): An individual marker uniquely identifies a single individual in a given context and corresponds to an individual constant in formal logic. Real-life examples are the matriculation number of a student (i.e. [STUDENT: #32145] = 'the student with the matriculation number 32145'), the chassis number of a car, the serial number of a product etc.
- *Literal*: A literal identifies an individual on the basis of its form. Example: in *"John" is name* the quoted string "John" indicates the word *John*, corresponding to the concept [WORD: "John"]. Similarly strings of digits represent themselves, e.g. *3.14159* in *the number 3.14159 is called Pi*, corresponding to [NUMBER: 3.14159].

This list of referent types has been extended in a number of ways to enhance the expressive power of the notation (cf. WAY 1991: 110):

Kind of referent	**Example**	**English reading**
Existential	[CAT]or[CAT:*]	*a cat* or *some cat*
Individual	[CAT: #10872]	*the cat #10872*
Definite Reference	[CAT: #][9]	*the cat*
Named individual	[CAT: Muffy]	*Muffy* or *the cat Muffy*
Specified set	[CAT:{Muffy,Yojo}]	*Muffy and Yojo*
Generic set	[CAT: {*}]	*cats* or *some cats*
Counted generic set	[CAT: {*} @5]	*five cats*
Definite set reference	[CAT: {*}#]	*the cats*
Universal quantifier	[CAT: ∀]	*every cat*
Universal negative	[CAT: ~]	*no cat*
Universal plural	[CAT: {*}∀]	*all cats*
Universal negative plural	[CAT: {*}~]	*no cats*
Fuzzy quantifier	[CAT:{*}@many]	*many cats*

Table 1: Referents

2.3 Conceptual Relations

The concepts in a conceptual graph are connected by *conceptual relations* that indicate what kind of relationship holds between their referents. In the formal theory there is only one *primitive* (dyadic) relation called LINK. In a system with only this primitive relation the empirical content must obviously be contained in the type system. All other conceptual relations

[9] The symbol # in [CAT: #] is a *control mark.* Although it occurs in the referent field of a concept, it does not identify the individual referent directly, but triggers a search in the context for something that is known to both speaker and listener. In the concept [ROPE: #] the marker # is an *indexical referent* which indicates a specific rope whose identity is assumed to be known. Other examples of *indexical referents* include #I and #you for the pronouns *I* and *you*; #this and #that for the demonstrative pronouns *this* and *that* and #now for the time in the current context.

that might be needed in a specific domain must ultimately be defined in terms of LINK. For convenience a number of relation types have been defined as a "starter set" (SOWA 1992:13). These include

- Case relations or thematic relations identical to those of other frameworks like case grammar, government and binding theory or lexical functional grammar.
- Spatial relations
- Attributive relations
- Intersentential relations

Case relations or *thematic relations* indicate how states, processes, activities or actions, expressed by predicators are linked to other participants (or actants), expressed by the subject, objects, or other complements. Typical examples for thematic roles of this sort are agent (AGNT), patient (PTNT), theme (THME), experiencer (EXPR), recipient (RCPT), instrument (INST), destination (DEST) and result (RSLT). In the ubiquitous example of an "opening act": *The man opened the door with a key* (the entity expressed by) the phrase *the man* is the agent, *the door* is the object (patient) affected by the act and *a key* the instrument which is used in the act (Figure 3).

[OPEN] —
(AGNT) → [MAN: #]
(PTNT) → [DOOR: #]
(INST) → [KEY: *].

Figure 3.

Spatial relation types represent spatial relationships that hold between two or more objects. These relation types include location (LOC) and a number of more specific relations such as IN, ON and ABOV, corresponding to spatial prepositions (*in, on, above*). For example, the sentence *There is man at the door* would be represented by the graph [MAN] → (LOC) → [DOOR: #].

Attributive relations indicate that an object has a certain property. These include the more general attribute relation (ATTR) as well as more specific relations such as part-of (PART) or characteristic (CHRC), where CHRC is used for inherent properties of an entity. The sentence *The ball is red*, for example, can be represented by the CG [BALL: #] → (CHRC) → [COLOR: red].

Intersentential relations express relationships between sentences (or propositions) and are comparable to conjunctions such as *because, after, before* or logical operators such as *and* and *or*. These relations link concepts of a special type called CONTEXT which have one or more conceptual graphs as their referent.

2.4 Contexts

The large box in Figure 1 represents a *context* containing the conceptual graph in Figure 4. This is actually an abbreviation for a concept of type SITUATION described by the graph in its referent field: [SITUATION: <graph>].[10] *Contexts* are a special kind of concept type, whose referent field may consist of several graphs. Contexts are useful for describing situations or propositions. Thus the types PROPOSITION and SITUATION are subtypes of CONTEXT. Contexts are needed to form collections of one or more graphs such as *schemata*, *prototypes* or *scripts*. They permit the nesting of conceptual graphs to express negation, modality, tense, and propositional attitudes such as belief, certainty, or uncertainty.[11] For the problem under consideration contexts are important, because they permit the representation of the background information that may come into play in the interpretation of *ad-hoc* compounds.

[CUT] –
(AGNT) → [PERSON: John]
(PTNT) → [ROPE: #]
(INST) → [KNIFE].

Figure 4.

[10] More precisely, it represents a SITUATION which is described by a PROPOSITION stated by a specific GRAPH: [SITUATION] → (DSCR) → [PROPOSITION] → (STMT) → [GRAPH] (cf. SOWA 1992:13ff.).

[11] The type field of CONTEXT is often omitted, so we just put brackets around to mark a context, e.g.: (PAST) → [[PERSON: John] ← (AGNT) ← [OPEN] → [DOOR: #]].

3 Type hierarchy

Concept types are organized in a hierarchy called a *type hierarchy* according to degrees of generality. Figure 5 shows a partial type hierarchy with the *universal type* ⊤ at the top and the general types ENTITY and SITUATION as proper subtypes. A type hierarchy is a partially ordered set of *type labels* which are specified as either *primitive* or *defined*. The partial ordering is achieved by the *subtype* relation "≤" ("<" for *proper subtype,* and its inverses "≥" for *supertype*, and ">" for *proper supertype*). The subtype relation is transitive, i.e. from PROCESS < EVENT and EVENT < SITUATION we can conclude that PROCESS < SITUATION. Every type of concept or relation in the semantic network organized around the type hierarchy may be associated with information structures such as *type definitions*, *canonical graphs*, *schemata*, and *prototypes*.

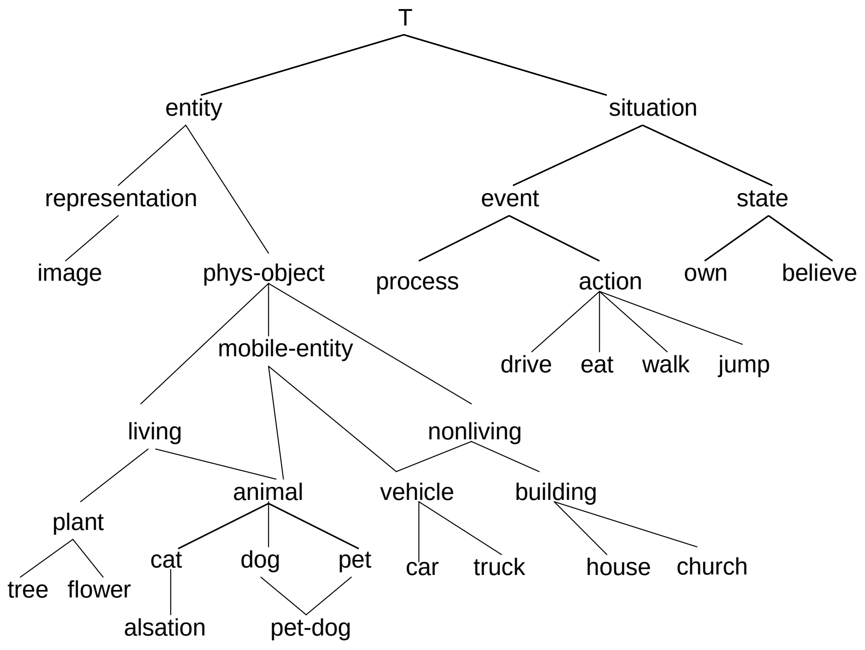

Figure 5. A partial type hierarchy (cf. SOWA 1992:8)

3.1 Canonical graphs

Every concept and relation type in the semantic network is associated with a conceptual graph that specifies the selectional constraints expected for the types of concepts and relations it contains. Such a graph is called a *canonical graph*. The case frames of FILLMORE'S case grammar (FILLMORE 1968) are examples of canonical graphs. Figure 6 shows one of FILLMORE'S frames written in conceptual graph notation. This graph specifies that OPEN requires an animate agent, any entity as its patient, and a physical object as instrument.

```
[OPEN] –
      (AGNT)→[ANIMATE]
      (PTNT) → [ENTITY]
      (INST) →[PHYS-OBJ].
```

Figure 6. Canonical graph

A graph is *canonical* if it is the representation of an observable situation. It is also canonical if it is the representation of an imaginable situation derivable from observable situations or gained by insight or creativity. The set of all such graphs in the lexicon, called the *canonical basis* or *canon,* determines selectional constraints. By definition, all graphs in the canon are canonical. All other canonical graphs are derivable from the canon by means of *canonical formation rules* (see below). Any graph derived by these rules will observe the constraints in the canon. Canonical graphs are not definitions. They only represent the structural constraints that must be obeyed to yield semantically well-formed sentences. In general, a type definition (see below) is much more detailed than a canonical graph.

3.2 Natural types and role types

Subtypes of ENTITY can be classified as *natural types* (PLANT, CAT, or BUILDING) or *role types* (PET, TEACHER, or CHILD), a classification that can account for the distinction between "relational" and "substantival" constituents made by Wildgen in his first assumption.

An individual is an instance of a *natural type* if it can be identified as belonging to this type on the basis of its inherent attributes and characteristics. Thus a cat can be recognized as such with reference to its observable features, similarly with a dog or a person.

An individual can only be identified as belonging to a role type by virtue of its relationship to other entities and situations. The type PERSON, for example, is a natural type, whereas TEACHER is a subtype of PERSON in the role of teaching.

```
TEACHER < PERSON
[TEACHER] —
        (AGNT) ← [TEACH] —
        (RCPT) → [ANIMATE]
        (PTNT) → [SUBJECT-MATTER].
```

Figure 7. Canonical graph of the role type TEACHER

Other examples of role types include CHILD < PERSON, PET < ANIMAL, PARENT < PERSON, FOOD < PHYSICAL-SUBSTANCE, TOOL < ENTITY.

Each of these role types is associated with a canonical graph specifying the implicit pattern of relationships it enters into. A child is a role played by one person with respect to another person who plays the role of parent; and conversely. A pet is an animal owned by some person and treated with care and affection. Food is a substance that people or animals eat or drink. A tool is an entity that plays the role of instrument for an act.

The distinction between natural types and role types is not always that clear-cut, because very often an object can play a certain role by virtue of its inherent properties. For example, a hammer is normally recognizable by its characteristic shape and can thus be regarded as a natural type. On the other hand it serves as a specific tool.

3.3 Canonical formation rules

All operations on conceptual graphs are based on combinations of six *canonical formation rules*, each of which performs one basic graph operation making a CG more specialized, more generalized, or transforming it while leaving it logically equivalent to the original.

The first two rules, which are illustrated in Figure 8, are *copy* and *simplify*. At the top is a CG for the sentence *(The person) John is cutting a rope.* The down arrow represents the copy rule. One application of this rule copies the AGNT relation, and a second application copies the subgraph → (PTNT) → [ROPE]. Both copies are redundant, since they add no new information. The up arrow represents two applications of the simplify rule, which performs the inverse operation of erasing redundant copies.

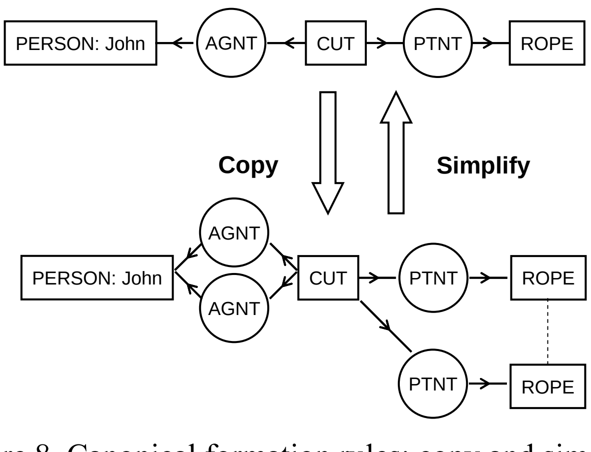

Figure 8. Canonical formation rules: copy and simplify

To show that the two copies of [ROPE] are *coreferent* (refer to the same individual), they are connected with a broken line, called a *coreference link*. Alternatively, coreference can be indicated by variables in the referent field preceded with an asterisk. The conceptual graphs [PERSON: John] – – – [TEACHER] and [PERSON: John *x] [TEACHER: *x] are equivalent.

Figure 9 illustrates the rules that restrict and generalize. At the top is a CG for the sentence *A person is cutting some physical object.* This is transformed to the CG for *The person John is cutting a rope* by first restricting the concept [PERSON], which represents some indefinite person, to the more specific concept [PERSON: John], which represents an individual person named John (*restriction by referent*). The second step is a *restriction by type* of the concept [PHYSICAL-OBJECT] to a concept of the subtype [ROPE]. Two applications of the generalize rule perform the inverse transformation of the bottom graph to the top graph.

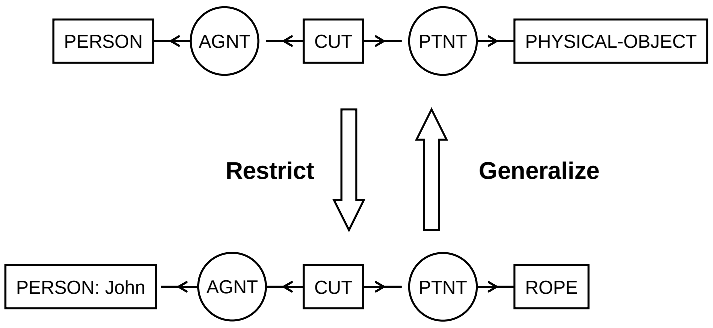

Figure 9. Canonical formation rules: restrict and generalize

Figure 10 illustrates the *join* and *detach* rules. At the top are two CGs for the sentences *John is cutting a rope* and *A rope is thick.* The join rule unifies the two identical copies of the concept [ROPE] to form a single CG for the sentence *John is cutting a thick rope.* The detach rule performs the inverse operation.

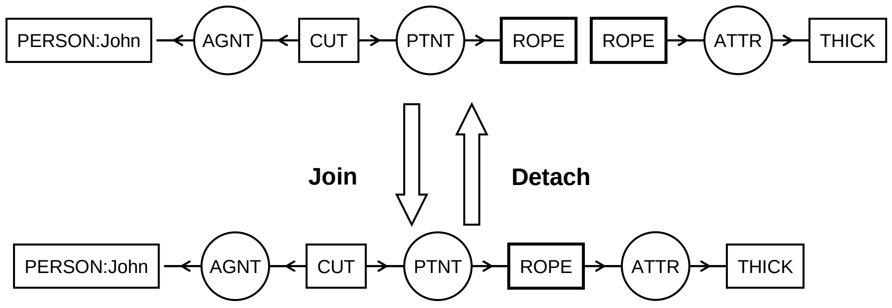

Figure 10. Canonical formation rules: join and detach

3.4 Type definitions

Concept and relation types are either *primitive* types that cannot be defined or types that are *defined* by *lambda abstractions*. A lambda abstraction can be derived from a conceptual graph by replacing one or more referents by variables bound by the lambda operator λ. For example, from [PERSON: John] ← (AGNT) ← [CUT] → (PTNT) → [ROPE] we can derive the monadic abstraction (λx) [PERSON: *x] ← (AGNT) ← [CUT] → (PTNT) → [ROPE]. This can also be expressed by putting the λ in the referent field of the concept in question: [PERSON:λ] ← (AGNT) ← [CUT] → (PTNT) → [ROPE], which represents the property of a person cutting a rope. Lambda abstractions can be used in various ways in the theory:

- *Definitions:* Monadic abstractions are used to define concept types, and *n*-adic abstractions are used to define *n*-adic relations types. They specify necessary and sufficient conditions.

- *Nameless types:* Instead of a permanently defined type label, the type field of a concept may contain a lambda abstraction that is created for temporary use. Nameless types can be used for restrictive modifications such as *a knife with a sharp blade*, which can be represented by [[KNIFE:λ] → (PART) → [BLADE] → (ATTR) → [SHARP]:*].
- *Schemata:* Like type definitions, schemata are defined by lambda abstractions. However, schemata differ from type definitions in that they only specify default values and expectations.
- *Prototypes:* Lambda abstractions also play a role in the characterization of prototypes representing a typical individual.

3.4.1 CONCEPT TYPES

A *concept type definition* specifies that a type label *t* is defined by a monadic abstraction (λ*x*) *G*, where *G* is a canonical graph. This can also be expressed by the notation **type** *t(x)* **is** *G*. The body *G* is called the *differentia* of *t*, and the type label of *x* is called the *genus* of *t*.

Any generic concept in a given canonical graph can be selected as the genus of a type definition. Given the canonical graph [PERSON] ← (AGNT) ← [CUT] → (PTNT) → [ROPE] with the generic concepts [PERSON], [CUT], and [ROPE] we can derive three different type definitions by lambda abstraction, only two of which really make sense:

type ROPE-CUTTER (x) **is**

    [PERSON:*x] ← (AGNT) ← [CUT] → (PTNT) → [ROPE]

This new type will enter the type hierarchy as a subtype of PERSON: ROPE-CUTTER < PERSON

type CUTTING-ROPE (x) **is**

    [PERSON] ← (AGNT) ← [CUT] → (PTNT) → [ROPE:*X]

CUTTING-ROPE < ROPE[12]

type rope-cutting (x) **is**

    [PERSON] ← (AGNT) ← [CUT:*X] → (PTNT) → [ROPE]

ROPE-CUTTING < CUT.

If we include the instrument we can also derive a type ROPE-CUTTING-KNIFE as a subtype of KNIFE.

type ROPE-CUTTING-KNIFE (x) **is**

```
[CUT] –
        (AGNT) → [PERSON]
        (PTNT) → [ROPE]
        (INST) → [KNIFE:*x]
```

A type definition simply assigns a label to a λ-expression to represent it in specific contexts such as the type field of a concept. Consequently, the type label in any concept can be replaced by the λ-expression that defines it, whose internal structure then becomes available for further processing if the need should arise. This is called *type expansion*.

3.4.2 RELATION TYPES

As has already been pointed out, the formal theory of conceptual graphs contains only one *primitive* relation type called LINK. This is a dyadic relation type from which others can be derived by *relational definitions*.

[12] This is similar in structure to *eating apple* but somewhat odd.

A relational definition, **relation** *t*(x_1 ... x_n) ***is*** *G*, specifies that the type label *t* for a conceptual relation is defined by the n-adic abstraction (λ x_1 ... x_n) *G*. The body *G* is called the *relator* of *t*. For example, the relation type AGNT may be defined in terms of a concept of type AGENT as follows:

relation AGNT(x, y) **is**

[ACT:*x] ← (LINK) ← [AGENT] → (LINK) → [ANIMATE:*y]

The number of concepts required by a relation type is called its *valence*. The *signature* of a relation type is the sequence of the most general concept types to which it can be linked. Since the AGNT-relation links an ACT to an ANIMATE, its signature is <ACT, ANIMATE>. The signature poses restrictions on the use of a relation. A relation can only be linked to concepts whose types are either identical to those specified in the signature or are subtypes of them. Thus the conceptual graph [STONE] ← (AGNT) ← [CRY] in its literal meaning is not semantically well-formed, because while CRY is a subtype of ACT, STONE is not a subtype of ANIMATE.

Although the primitive relation type LINK is dyadic, we can define relation types of arbitrary adicity. The conceptual graph represented in Figure 1 contains the monadic relation type PAST which is defined in terms of the dyadic relation types point of time (PTIM), successor (SUCC) and the contextually defined time #now as follows:

relation PAST(x) **is**

[SITUATION: *x] → (PTIM) → [TIME] → (SUCC) → [TIME: #now].

Relational expansion is the counterpart of type expansion. Given the relational definition:

relation BUILD(x, y) **is**

[PERSON: *x] ← (AGNT) ← [MAKE] → (RSLT) → [BUILDING: *y] or equivalently

BUILD = [PERSON: λ_1] ← (AGNT) ← [MAKE] → (RSLT) → [BUILDING: λ_2],

the type label build in [PERSON: John] ← (BUILD) → [HOUSE:*] can be replaced by its definition yielding

[PERSON: John] ← ([PERSON: λ_1] ← (AGNT) ← [MAKE] → (RSLT) → [BUILDING: λ_2]) → [HOUSE:*],

which can be simplified to [PERSON: John] ← (AGNT) ← [MAKE] → (RSLT) → [HOUSE: *] by a form of λ-conversion.

3.5 Schemata and Prototypes

Whereas *types* are defined by stating the necessary and sufficient conditions that distinguish them from other types, which is formally achieved by canonical graphs specifying a supertype (*genus proximum*) and the distinguishing concepts and relations (*differentia specifica*), *schemata* specify the concepts and relations that are commonly associated with a certain concept type. Schemata go beyond type definitions in that they incorporate *domain-specific knowledge* about the typical constellations of entities, attributes, and events in the real world. A concept type may have at most one definition, but arbitrarily many schemata.[13]

A **prototype** is a specialisation of a type by a collection of one or more schemata which show the form of a typical representative. Prototypes exhibit default values that are valid for a typical case, but not necessarily for a particular case.[14]

[13] The set of related schemata associated with a concept is called a *schematic cluster*.

[14] Although metaphors are important for an analysis of compounding I will not consider them here. Metaphors appear to violate the rules for combining canonical graphs. However, WAY (1991) developed a technique for handling metaphors by dynamically extending the type hierarchy.

4 Conceptual graphs in action

4.1 The compositionality principle

As has been pointed out in the beginning, the interpretation of *non-lexicalised* compounds presupposes some version of the principle of compositionality, assuming that the meaning of an expression is derivable from the meanings of its constituent elements together with its syntactic structure and surrounding context.

The close relationship between syntax and semantics can be illustrated with sentences that are structurally ambiguous:

(1) John talked to the man in the garden

In this sentence the reference of the local prepositional phrase *in the garden* is ambiguous:

(2) $[[\text{John}]_{NP}$ $[\text{talked}$ $[\text{to the man}]_{PP}$ $[\text{in the garden}]_{PP}]_{VP}]_{S}$

(3) $[[\text{John}]_{NP}$ $[\text{talked}$ $[\text{to}$ $[\text{the man}$ $[\text{in the garden}]_{PP}]_{NP}]_{PP}]_{VP}]_{S}$

These different syntactic structures correspond to different semantic structures. In terms of conceptual graphs (4) is the reading corresponding to the analysis in (2) and (5) is the reading of analysis (3):

(4) *in the garden* specifies the location (LOC) of *talk*

[TALK] –
  (AGNT) → [PERSON:John]
  (RCPT) → [MAN:#]
  (LOC) → [GARDEN:#].

(5) *in the garden* specifies the location of *the man:*

[PERSON:John] ← (AGNT) ← [TALK] → (RCPT) → [MAN:#] → (LOC) → [GARDEN:#].

The semantic representation (in terms of conceptual graphs) is incrementally derived by joining the conceptual graphs associated with each constituent under the (partial) control of the syntactic rules. In the present context these are assumed to be based on some version of the $\overline{X}$ -schema, preferably with binary branching, augmented by arguments representing conceptual graphs and operations that perform a join on these graphs. We will assume the following framework:

1. A lexicon that associates each word with a concept type and one or more canonical graphs. In the case of natural types the graph may consist of a single concept. With verbs, adjectives and role types the associated canonical graphs can be quite complex. Concept types may be primitive or defined, in which case the type definition may come into play. Concept types may be associated with schemata and prototypes to account for background knowledge.
2. Every canonical graph associated with a node in the syntax tree contains a concept marked as its head concept. The head concept is the starting point for the unification of this graph with the graphs of other nodes.
3. Each phrase structure rule has the general form $X^n(G) \rightarrow Y(G_1)\ X^{n-1}(G_0)\ Z(G_2)$, $\{G = join(G_0, G_1, G_2)$, where Y and Z may be empty. If both Y and Z are empty, $G = G_0$.
4. The canonical graph of category X^{n-1} is formed by unifying the head concepts of the categories Y and Z either with the head concept of X^{n-1} or with another concept in the graph associated with X^{n-1}. The head concept of X^n is identical with that of X^{n-1}.
5. If there are more alternatives for the unification of the head concept with the concepts of the other graphs, the decision may be controlled by the syntax rule. The selection of the thematic role of the subject may follow a thematic hierarchy like (AGNT < EXPR < INST < PTNT < … etc.).
6. Syntactic ambiguities may be resolved by constraints that hold for the unification of conceptual graphs.
7. More complex conceptual structures may be built on the basis of schemata and prototypes.

Unfortunately, the limited space available does not permit demonstrating this system *in extenso*. However, the general principles can be explained satisfactorily even with simple phrases such as *a passionate girl** vs. *a passionate teacher**.[15] After all, the syntax of nominal compounds is very simple. Although these phrases have the same phrase structure (see Figure 11), they obviously differ in meaning. In the phrase *a passionate girl* the adjective *passionate* refers to the attribute of a person, whereas in the most natural interpretation of *a passionate teacher* it refers to the manner of teaching. This can be accounted for in the following way:

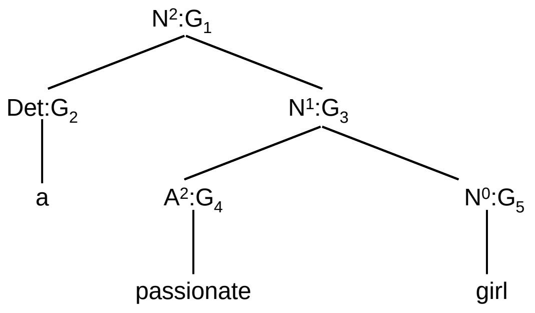

Figure 11. Noun phrase structure

The adjective *passionate* has two canonical graphs associated with it:

1. [PASSIONATE] ← (ATTR) ← [PERSON]
2. [PASSIONATE] ← (MANR) ← [ACT]

The graph associated with the noun *girl* is simply the concept [GIRL]. Since GIRL < PERSON, the most likely reading of *passionate* is the first one. Thus we can assume the following equations:

G_4 = [PASSIONATE] ← (ATTR) ← [PERSON] [GIRL*] = G_5

$G_3 = G_4 \sqcup G_5$, where $\sqcup$ symbolizes the "maximal join" operator.[16]

In order to be able to perform the join, the concept [PERSON] must be type-restricted to [GIRL], which is permissible since GIRL < PERSON:

G'_4 = [PASSIONATE] ← (ATTR) ← [GIRL].

Since the concept node in G_5 has no relations attached to it, it can be merged with the concept node [GIRL] in G'_4:

$G_3 = G'_4$ = [PASSIONATE] ← (ATTR) ← [GIRL*]

The canonical graph associated with the indefinite article in this context is the generic concept [⊤:*], where ⊤ is the universal concept type. Merging this with the head concept in G_3 by type-restricting ⊤ to girl (GIRL <) adds no information, hence $G_1 = G_3$.

```
[TEACH] –
        (AGNT) → [TEACHER*]
        (THME) → [SUBJECT-matter]
        (RCPT) → [PERSON].
```

Figure 12. Canonical graph for TEACHER

The noun *teacher* in *a passionate teacher* is a role type and associated with a more elaborate canonical graph specifying its role (see Figure 12). Here we have a potential ambiguity between *passionate as a person* and *passionate as a teacher*. Since TEACHER < PERSON the concept [PERSON] in the first reading of *passionate* can be type-restricted to [TEACHER] and unified with the agent in Figure 12.

```
[TEACH] –
        (AGNT) → [TEACHER*]   [TEACHER] → (ATTR) → [PASSIONATE]
        (THME) → [SUBJECT-matter]
        (RCPT) → [PERSON].
```

Figure 13: Joining two graphs

[15] Here the asterisk is used to mark the head of the phrase.

[16] Informally we can define a maximal join as an operation that derives the simplest CG from a set of CGs with a maximal number of join operations.

The join is achieved by deleting the identical concept node [TEACHER] and attaching the dependent subgraph → (ATTR) → [PASSIONATE] to the head node. The result is shown in Figure 14

[TEACH] –
(AGNT) → [TEACHER*] → (ATTR) → [PASSIONATE]
(THME) → [SUBJECT-MATTER]
(RCPT) → [PERSON].

Figure 14: Conceptual graph for *a passionate teacher*

In the more natural reading *passionate* modifies the act of teaching.

[PASSIONATE] ← (MANR) ← [ACT]
[TEACH] –
(AGNT) → [TEACHER*]
(THME) → [SUBJECT-MATTER]
(RCPT) → [PERSON].

Figure 15: Canonical graphs for *passionate* and *teacher*.

Since TEACH < ACT the concept [ACT] can be type restricted to [TEACH], permitting the join in Figure 16. This can be done by deleting the first of the identical concept nodes [teach] and attaching the dependent subgraph [passionate] ←(manr) ← to the remaining node (Figure 16

[PASSIONATE] ← (MANR) ← [TEACH] –
(AGNT) → [TEACHER*]
(THME) → [SUBJECT-matter]
(RCPT) → [PERSON].

Figure 16: Canonical graph for *passionate teacher*

Figure 17 is a visual representation of the bottom-up derivation under the control of the phrase structure. The canonical graphs associated with the indefinite article and the words *passionate* und *teacher* in the lexicon are assigned to the nodes *Det*, A^2 and N^0, respectively. In the next step the join of these two graphs is assigned to the node N^1. And finally this is joined with the graph associated with *Det* and assigned to the node N^2.

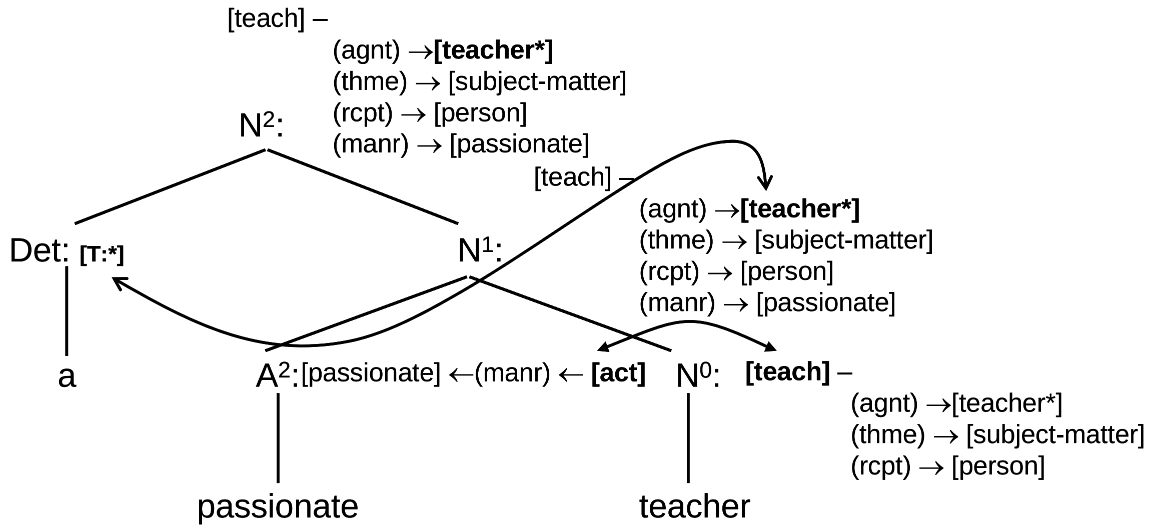

Figure 17: A conceptual graph derivation

4.2 The compositionality of nominal compounds

The derivation of the meaning of compounds like *silver spoon* vs. *soup spoon* is quite analogous to that of phrases. Since their syntax is very simple (see Fig. 18[17]), there is no syntactic ambiguity either. Nevertheless, their interpretation is quite different.[18] Again the different interpretation of *silver spoon* vs. *soup spoon* can be explained with the different canonical graphs associated with their constituents.[19] For example, *silver* (concept [SILVER], SILVER < METAL) can be characterized as the material used in the production of an object. This can be accounted for by the canonical graph [SILVER] ← (MATR) ← [MAKE] → (RSLT) → [PHYS-OBJ]. A canonical graph for *spoon* (concept [SPOON]) would have to contain the information that it is a tool (SPOON < TOOL, relation INST), which is used to ingest liquid (or pulpy) food: [SPOON] ← (INST) ← [INGEST] → (PTNT) →[FOOD] →(ATTR) → [LIQUID-OR-PULPY].[20] The concept [SOUP] is compatible with "liquid (cooked) food".

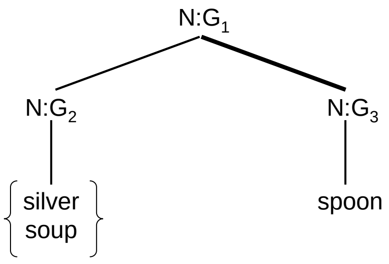

Figure 18: Compound structure

For *silver spoon* the exact semantics of *spoon* does not really matter. It is sufficient to assume that SPOON is a subtype of PHYS-OBJ.

G_2 = [SILVER] ← (MATR) ← [MAKE] → (RSLT) → [PHYS-OBJ] [SPOON*] = G_3

Type restriction of PHYS-OBJ to SPOON produces G'_2:

G'_2 = [SILVER] ← (MATR) ← [MAKE] → (RSLT) → [SPOON*]. This trivially joins with G_3 to yield $G_1 = G'_2$.

The derivation of the semantics of *soup spoon* is more complicated. *Spoon* can be considered both a natural type and a role type. For the interpretation of the compound the information related to its role is more relevant. Other information that may be contained in the s*chematic cluster* associated with the type, e.g. the fact that a spoon consists of a "shallow bowl with a handle", can be added if required. Assuming for the sake of the argument a very simple type definition for *soup* like SOUP = [FOOD:λ] → (ATTR) → [LIQUID], i.e. ignoring the fact that soup is made by boiling certain ingredients in water or stock, the basic meaning of the compound can be derived as follows:

G_2 = [SOUP]

G_3 = [SPOON] ← (INST) ←[INGEST]→(PTNT)→ [FOOD]→(ATTR)→[LIQUID-OR-PULPY]

G'_2 = [FOOD] → (ATTR) → [LIQUID] from G_2 by type expansion

G'_3 = [SPOON] ← (INST) ←[INGEST]→(PTNT)→ [FOOD]→(ATTR)→[LIQUID] by type restriction since LIQUID < LIQUID-OR-PULPY

$G'_1 = G'_2 \sqcup G'_3 = G'_3$

G_1 = [SPOON] ← (INST) ←[INGEST]→(PTNT)→ [SOUP] from G'$_1$ by *type contraction.*

[17] The bold edge indicates the projection line identifying the syntactic head of the construction.

[18] Obviously *soup spoon* and *silver spoon* are not really *ad-hoc* compounds. However, what counts here is that they can be regarded as compositionally derivable.

[19] In this context the exact status of these canonical graphs, i.e. whether they are mere statements of selectional restrictions, or schemata, or result from type expansions, will not be elaborated.

[20] The concept type LIQUID-OR-PULPY is a derived type formed by the union LIQUID ∪ PULPY and is thus a common supertype of both, i.e. LIQUID ∪ PULPY > LIQUID, PULPY.

5 Conclusion

Due to the limited space available, the application of the system of conceptual graphs outlined in the major part of this paper to the problem domain adressed in WOLFGANG WILDGEN's study has admittedly been rather sketchy. In particular, it has not been shown explicitly how background information can be integrated into the interpretation process. What is important, though, is the fact that the relevant information structures – conceptual schemata, prototypes, scripts, discourse structures – can all be represented as conceptual graphs and can be handled by the same operations. I hope that I have at least been able to demonstrate the potential of the theory of conceptual graphs.

6 References

EVENS, MARTHA WALTON (ED.)

1988 *Relational models of the lexicon: Representing knowledge in semantic networks.* Cambridge University Press: Cambridge (England) [u.a.]

FANSELOW, GISBERT

1981 *Zur Syntax und Semantik der Nominalkomposition: Ein Versuch praktischer Anwendung der Montague-Grammatik auf die Wortbildung im Deutschen.* Max Niemeyer Verlag: Tübingen (= Linguistische Arbeiten; 107)

FILLMORE, CHARLES J.

1968 The case for case. In: Fillmore (2003), 23–122

2003 *Form and Meaning in Language. Volume I: Papers on Semantic Roles.* CSLI Publications: Stanford (Cal.)

HÜLLEN, WERNDER

1976 *Linguistik und Englischunterricht 2: Didaktische Analysen.* Heidelberg: Quelle & Meyer

KÜRSCHNER, WILFRIED

1974 *Zur syntaktischen Beschreibung deutscher Nominalkomposita: Auf der Grundlage generativer Transformationsgrammatiken.* Niemeyer: Tübingen (= Linguistische Arbeiten; 18)

MEYER, RALF

1993 *Compound Comprehension in Isolation and in Context: The contribution of conceptual and discourse knowledge to the comprehension of German novel noun-noun compounds.* Niemeyer: Tübingen (= Linguistische Arbeiten 299)

MINEAU, GUY W., BERNARD MOULIN & JOHN F. SOWA (EDS.)

1993 *Conceptual Graphs for Knowledge Representation.* Springer-Verlag: Berlin [u.a.] (= Lecture Notes in AI; 699)

NAGLE, TIMOTHY E., JANICE A. NAGLE, LAURIE L. GERHOLZ, PETER W. EKLUND (EDS.)

1992 *Conceputal Structures: Current Research and Practice.* Ellis Horwood: New York [u.a.]

PEIRCE, CHARLES SANDERS

1909 Existential Graphs. MS 514 with commentary by John F. Sowa
[URL: http://www.jfsowa.com/peirce/ms514.htm]

PUSTEJOVSKY, JAMES (ED.)

1993 *Semantics and the Lexicon.* Kluwer Academic Publishers: Dordrecht

REINHARD, SABINE

2001 *Deverbale Komposita an der Morphologie-Syntax-Semantik-Schnittstelle: ein HPSG Ansatz.* Diss. Tübingen. [URL: http://w210.ub.uni-tuebingen.de/dbt/volltexte/2002/466]

ROBERTS DON D.

1973 *The Existential Graphs of Charles S. Peirce.* Mouton: The Hague

SHIN, SUN-JOO

2002 *The Iconic Logic of Peirce's Graphs.* The MIT Press: Cambridge (Mass.) - London

SOWA, JOHN F.

1984 *Conceptual Structures: Information Processing in Mind and Machine.* Addison Wesley: Reading (Mass.) [u.a.]

1988 Using a lexicon of canonical graphs in a semantic interpreter. In: EVENS (1988), 114–137.

1991 Toward the expressive power of natural language. In: SOWA (ed.) (1991), 157–189

1992 Conceptual Graphs Summary. In: NAGLE et al. (1992), 3–51

1993a Relating diagrams to logic. In: MINEAU et al. (1993), 1–35

1993b Lexical Structures and Conceptual Structures. In: PUSTEJOVSKY (1993), 223–262

2000 *Knowledge Representation: Logical, Philosophical, and Computational Foundations.* Brooks/Cole: Pacific Grove [u.a.]

SOWA, JOHN F. (ED.)

1991 *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. Morgan Kaufmann Publishers: San Mateo (Cal.)

SOWA, JOHN F. & EILEEN C. WAY

1986 Implementing a semantic interpreter using conceptual graphs. In: *IBM Journal of Research and Development* **30**:57–69

WAGNER, KARL HEINZ

1971 Zur Nominalisierung im Englischen. In: ARNIM VON STECHOW (ed.), *Beiträge zur generativen Grammatik. Referate des 5. Linguistischen Kolloquiums Regensburg.* 1970. (Schriften zur Linguistik; 3) Vieweg: Braunschweig, 1971, 264–272.

WAY, EILEEN CORNELL

1991 *Knowledge Representation and Metaphor.* Kluwer Academic Publishers: Dordrecht [u.a]

WILDGEN, WOLFGANG

1982 Zur Dynamik lokaler Kompositionsprozesse: Am Beispiel nominaler ad hoc-Komposita im Deutschen. In: *Folia Linguistica* **16**:297–344