Handling Valency and Coordination in Database Semantics

Roland Hausser

Universität Erlangen-Nürnberg
rrh@linguistik.uni-erlangen.de

Abstract Database Semantics is designed to reconstruct the mechanism of natural language communication computationally in the form of a talking robot. For this, the traditional components of grammar, such as the lexicon, the morphology, the syntax, and the semantics must be reconstructed and integrated into a functioning agent-oriented system. After explaining the basic model of natural language communication, this paper reconstructs the notions of Valency Theory in Database Semantics. It also shows, how the grammatical relations of valency are to be combined with the grammatical relations of coordination.[1]

1 Sign-based vs. agent-based approaches to language

Most linguistic approaches are *sign-oriented* in that they analyze expressions of natural language as objects, fixed on paper, magnetic tape or by electronic means. They abstract away from the aspect of communication and analyze signs as hierarchical structures which are represented as trees and formally based on the principle of *possible substitutions*.

Database Semantics (DBS), in contrast, is *agent-oriented* in that it analyzes signs as the result of the speaker's language production and as the starting point of the hearer's language interpretation. Inclusion of the agents' production and interpretation procedures requires in a time-linear analysis which is formally based on the principle of *possible continuations*.

The goal of Database Semantics is a theory of natural language communication which is complete with respect to function and data coverage, of low mathematical complexity, and suitable for an efficient implementation on the computer. The central question of Database Semantics is

How does communicating with natural language work?

In the most simple form, DBS answers this question as follows:

Natural language communication takes place between cognitive agents. These have interfaces for non-verbal recognition and action at the level of context, and verbal recognition and action at the level of language. Each agent contains a database in which contents are stored. These contents consist of the agent's knowledge, memories, current

[1] This paper benefited from comments by JaeUn Choe (Korea University, Seoul), Besim Kabashi (Friedrich-Alexander-University, Erlangen), Haitao Liu (Communications University of China, Beijing), and Brian MacWhinney, (Carnegie Mellon University, Pittsburgh).

recognition, intentions, plans, etc. Contents are read into and out of the database by means of a time-linear algorithm. Cognitive agents can switch between the speaker- and the hearer-mode (turn-taking).

In a communication procedure, an agent in the speaker-mode codes content from its database into signs of language which are realized externally via the language output interface. These signs are recognized by another agent in the hearer-mode via the language input interface, their content is decoded, and stored in the second agent's database. This procedure is successful if the content coded by the speaker is decoded and stored equivalently by the hearer.

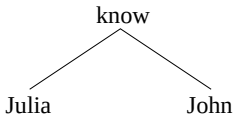
2 Complementation

In Valency Theory (cf. Ágel 2000) there is a basic distinction between valency carriers (actants) and valency fillers (complements). Valency carriers have certain slots for which there must be suitable fillers in order for a sentence to be grammatical and complete. For example, the verb know is a valency carriers with valency positions for two nominal fillers, one serving as the subject, the other as the object.

In Dependency Grammar, the relations between a valency carrier and its fillers in a sentence are shown in the form of a tree, called dependency graph or stemma. Consider the following dependency graph of the sentence Julia knows John:

2.1 Representing dependency relations as a tree

dependency graph (stemma)



Supplying a valency carrier with suitable fillers is called complementation.

In Database Semantics, the grammatical structure of a sentence is represented as a set of proplets rather than a tree. Proplets are non-recursive (flat) feature structures in the sense that attributes may not take feature structures as values.

2.2 Representing dependency relations as a set of proplets

sur:	sur:	sur:
noun: Julia	verb: know	noun: John
cat: nm	cat: decl	cat: nm
sem: sg	sem: pres	sem: sg
mdr:	arg: Julia John	mdr:
fnc: know	mdr:	fnc: know
idy: 14	prn: 6	idy: 15
prn: 6		prn: 6

The relations between the valency carrier *know* and its fillers *Julia* and *John* is coded by values for certain attributes. More specifically, the proplet *know* has the attribute

arg with the values Julia John, and the proplets *Julia* and *John* have the attribute fnc with the value know (bidirectional pointering).

Proplets belonging to the same proposition share a common proposition number (here prn: 6). Each proplet is an autonomous item which may be stored anywhere in memory without affecting the coding of grammatical relations. Thus, the grammatical relations characterized by sign-oriented approaches in the form of a tree are recoded equivalently in Database Semantics by means of values of certain attributes. They establish a bidirectional pointering between proplets of valency carriers and their filler proplets, regardless of where they are located in storage.

There is a total of eleven attributes in the proplets of example 2.2, the values of which have the following properties:

2.3 PROPLET ATTRIBUTES AND THEIR VALUES

1. Surface attribute: sur
   Each proplet has a unique surface attribute. Its value is the language dependent surface of a proplet, needed for lexical lookup in the hearer-mode. After lexical lookup, the sur value is usually omitted.

2. Core attributes: noun, verb, adj
   Each proplet has a unique core attribute, which gets its value from the lexicon. From a sign-theoretic point of view,[2] the core values may consist of a *concept*, a *pointer*, or a *marker*, which corresponds to the sign kinds of *symbol*, *indexical* and *name*. While a verb-attribute can only take a concept as its value, an adj-attribute can take a concept or a pointer. The most general kind is the noun-attribute, which can take a concept, a pointer, or a marker as value.[3]

3. Continuation attributes: fnc, arg, mdd, mdr[4]
   Each proplet has several continuation attributes, which get their values by copying during the composition of proplets (cf. 3.1). The values consist of characters (char), which represent the names of other proplets. In complete propositions, the values of fnc, arg, and mdd must be non-NIL, while that of mdr may be NIL.
   Additional continuation attributes are pc (previous conjunct) and nc (next conjunct) in verbal proplets. They are used for connecting propositions in a time-linear sequence.

4. SynSem attributes: cat, sem
   Each proplet has the SynSem attributes cat and sem. They get their values in part from the lexicon, for example nm (for name), and in part by copying.

5. Book-keeping attributes: prn (proposition number)
   Each proplet has one or more book-keeping attributes, which get their values by the control structure of the parser and consist of numbers (integers). In connected proplets, these values must be non-NIL.

---

[2] See FoCL, Chapter 6.

[3] Seventh Principle of Pragmatics (PoP-7). See FoCL, p. 107.

[4] For functor, argument, modified, and modifier, respectively.

Additional book-keeping attributes are idy (identity), wrn (word number), and trc (transition counter).

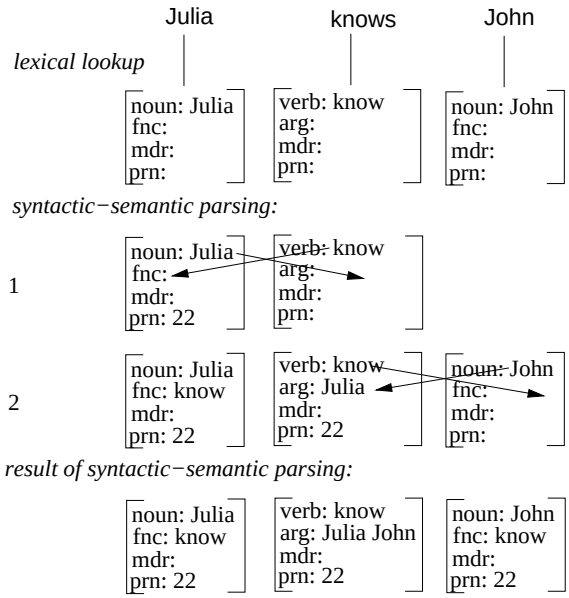
Coding grammatical relations in terms of attributes and values is suitable not only for the treatment of complementation, i.e. obligatory and optional relations between a carrier and its fillers which are restricted in some grammatical way (for example agreement), but also for the treatment of adjuncts, which are usually excluded by lexical approaches to Valency Theory (cf. Herbst 1999).

In other words, the handling of grammatical relations in Database Semantics applies not only to the traditional valency relations, but to functor-argument structures in general. Treating valency relations as an instance of functor-argument structure has several advantages, one of them being that valency relations are supplied with the standard semantic interpretation of functor-argument structure.

3 Basic model of communication in Database Semantics

Using the reconstruction of complementation in Database Semantics, we may illustrate the basic functioning of natural language communication. An agent in the hearer-mode receives a sequence of unanalyzed surfaces as input. During lexical lookup, these surfaces are matched with corresponding lexical proplets. Lexical proplets are *unconnected* in that the attributes coding relations to other proplets have no values yet.

3.1 CODING VALENCY STRUCTURE IN THE HEARER-MODE

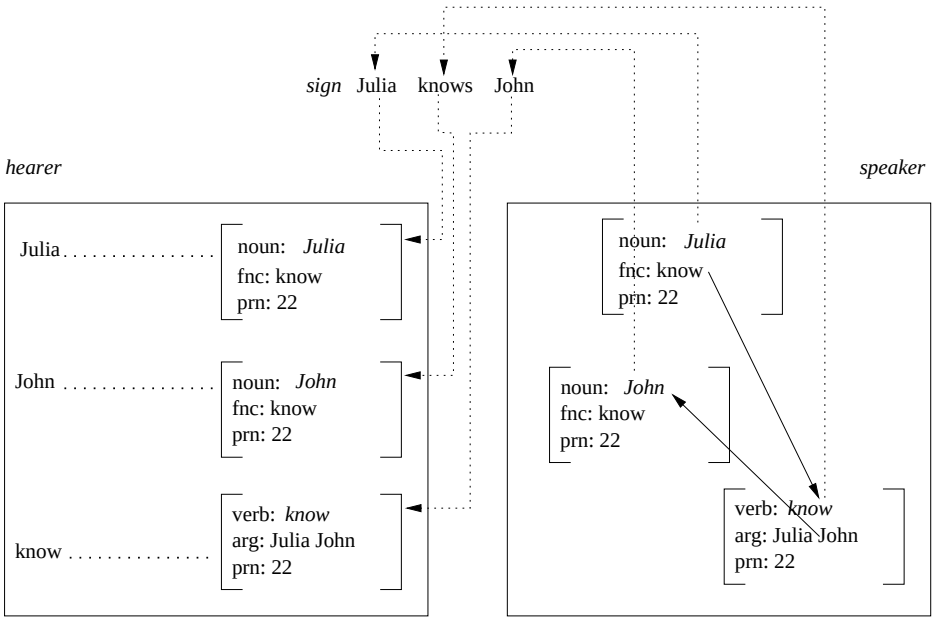


After lexical lookup, the proplets are connected by syntactic-semantic parsing. This strictly time-linear procedure is performed by an LA-grammar called **LA-hear**. It is

based on copying values (here indicated by arrows) and by providing values to the book-keeping attributes (here the prn value 22).

The result of syntactic-semantic parsing in the hearer-mode is a(n unordered) set of proplets. For purposes of indexing and retrieval, these proplets are stored at the end of alphabetically ordered token lines. Each token line begins with the name of the core value and is followed by all proplets containing this core value. In this way, the content coded by the natural language expression Julia knows John is stored in the database of the hearer (see *hearer* on the left):

3.2 TRANSFER OF CONTENT FROM SPEAKER TO HEARER



When the hearer turns into a speaker (see *speaker* on the right), stored content is activated by means of a time-linear navigation. Let us assume, for example, that *Julia* has been activated by the agent's control structure as the navigation-initial proplet. It has the attribute fnc with the value know and the attribute prn with the value 22. Based on these values, the continuation proplet *know* is being activated next (see arrow from *Julia* to *know*). The first value of its arg attribute, i.e. Julia, confirms the legality of the first navigation step. The second value, i.e. John, provides the information for continuing the navigation by activating a third proplet (see arrow from *know* to *John*).

Such a navigation through stored content serves as the basic model of *thought* in Database Semantics. Based on the grammatical relations between proplets (intrapropositional navigation) and between propositions (extrapropositional navigation), the navigation uses the standard retrieval mechanism of the database.

Because proplets normally provide more than one possible successor, the navigation algorithm, called **LA-think**, must make choices. The most basic solutions are either completely random choices or completely fixed choices, based on some predefined

schema. For rational behavior, however, the **LA-think** grammar must be refined into a control structure which chooses between continuation alternatives based on the evaluation of external and internal stimuli, the frequency of previous traversals, learned procedures, theme/rheme structure, etc.

For present purposes, we assume the predefined schema of a standard navigation, starting with the verb and continuing with the arguments in their given order. This navigation may be represented schematically as VNN, with V representing the verb proplet, the first N the subject, and the second N the object. In principle, any such navigation through the word bank is independent of language. However, in cognitive agents with language, the navigation serves as the speaker's *conceptualization*, i.e., as the speaker's choice of *what to say* and *how to say it*.

A conceptualization defined as a time-linear navigation through content makes language production relatively straightforward: If the speaker decides to communicate a navigation to the hearer, the concept names (i.e., values of the core attributes) of the proplets traversed by the navigation are translated into their language-dependent counterparts and realized as external signs. In addition to this language-dependent lexicalization of the universal navigation, the system must provide

1. language-dependent word order
2. function word precipitation
3. word form selection for proper agreement

This process is handled by language-dependent **LA-speak** grammars in combination with language-dependent word form production. For example, the word form ate is produced from an *eat* proplet the sem attribute of which contains the value past.

Because word form selection for proper agreement involves a large amount of morphosyntactic detail, the language-specific production from the VNN navigation of our example is characterized by the following simplified format:

3.3 SCHEMATIC PRODUCTION OF Julia knows John. (SPEAKER-MODE)

activated sequence	*realization*
i	
V	
i.1 n	n
V N	
i.2 fv n	n fv
V N	
i.3 fv n n	n fv n
V N N	
i.4 fv p n n	n fv n p
V N N	

The letter 'i' stands for the number of the sentence produced. The letters n, fv and p are abstract surfaces for *name*, *finite verb*, and *punctuation* (here full stop), respectively.

The derivation begins with a navigation from V to N, based on **LA-think**. In line i.1, the N is realized as the n 'Julia', and in line i.2 the V is realized as the fv 'knows' by

LA-speak. In line i.3, **LA-think** continues the navigation to the second N, which is realized as the n 'John' by **LA-speak**. Finally, **LA-speak** realizes the p '.' from the V proplet (line i.4).

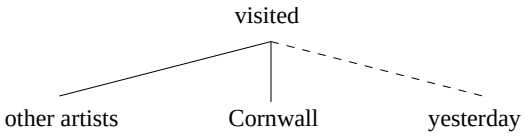
The time-linear switching between **LA-think** and **LA-speak** is motivated not only by psychological considerations,[5] but also by computational efficiency. The reason is that realizing the surfaces of proplets as soon as possible (instead of navigating to the end of the proposition first) results in a more restricted set of candidates for matching by the LA-speak rules than having to consider the proposition's complete set of proplets.

The method of production shown in 3.3, based on an underlying VNN navigation, can be used to realize not only an SVO surface (subject-verb-object) as in the above example, but also an SOV and (trivially) a VSO surface (Greenberg 1963). The ordering and lexicalization are specified by the rules of an **LA-speak** grammar, whereby the design of these grammars is supported conceptually by abstract derivations like 3.3.

4 Treating adjuncts

Having outlined the basic mechanism of natural language communication in Database Semantics with a reanalysis of complementation, let us turn next to the treatment of adjuncts. Herbst 1999 presents the following example, shown here in simplified form:

4.1 DEPENDENCY GRAPH (STEMMA) WITH TWO-PLACE VERB AND ADJUNCT



The optional character of the adjunct is indicated by the dotted line. In Database Semantics, the same example is represented as the following set of proplets:

4.2 CORRESPONDING REPRESENTATION IN DATABASE SEMANTICS

sur:	sur:	sur:	sur:	sur:
noun: artist	adj: other	verb: visit	noun: Cornwall	adj: yesterday
cat: pnp	cat: adn	cat: decl	cat: nm	cat: adv
sem: pl	sem:	sem: past	sem: sg	sem:
mdr: other	mdr:	arg: artist Cornwall	mdr:	mdr:
fnc: visit	mdd: 16 artist	mdr: yesterday	fnc: visit	mdd: visit
idy: 16	prn: 7	prn: 7	idy: 17	prn: 7
prn: 7			prn: 7	

[5] The *principle of incrementality* emphasizes the extent to which interpretation occurs in real time as new words are being heard (cf. MacWhinney 1987). The linkage of valency relations to incremental parsing found in the LA approach also provides a comprehensive approach to the aspects of the language learning problem known as the logical problem of language acquisition. In particular, MacWhinney 2004, 2005 has shown that children can induce the correct set of valency relations in their target language by focusing on meaningful relations in main clauses, along with their verbal complements.

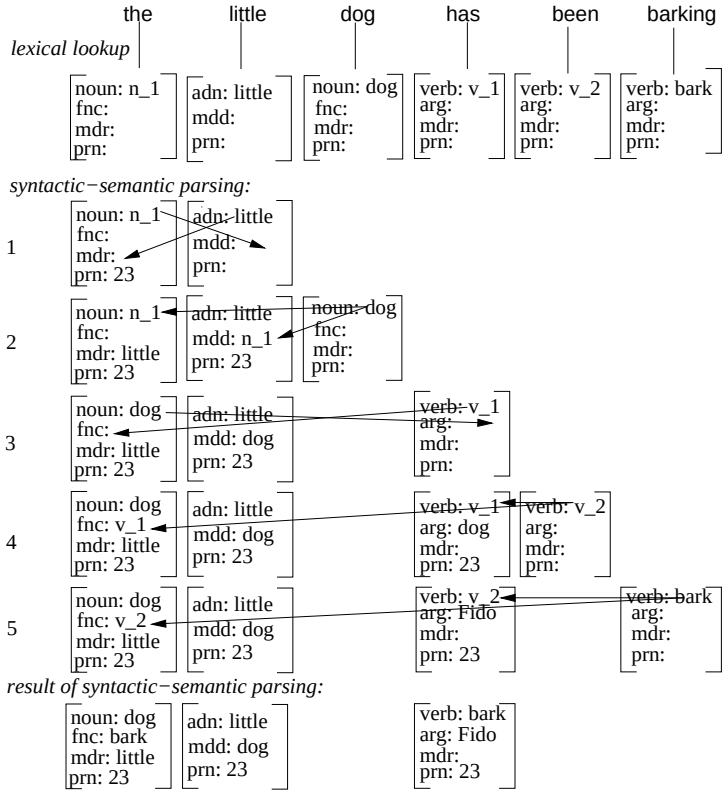
There are two adjuncts, the adnominal modifier other and the adverbial modifier yesterday. The bidirectional pointering is between the modified (mdd), i.e. artist and visit, and the modifier (mdr), i.e. other and yesterday, respectively. The optional character of modifiers is treated as a property of the mdr attribute (typing), which may have the value NIL.

5 The treatment of certain function words (translatives)

One of the basic distinctions in Tesnière 1959 is between *mot plein* and *mot vide*, which may be translated as content word and function word, respectively. Examples of function words are determiners and auxiliaries, which are analyzed as 'translatives'.

In Database Semantics, function words are fused with associated content words during interpretation in the hearer-mode (absorption) and extracted during production in the speaker-mode (precipitation). Consider the following example illustrating absorption of a determiner and two auxiliaries in a hearer-mode derivation:

5.1 PARSING The little dog has been barking IN THE HEARER MODE



Because Database Semantics is strictly surface compositional, all word form surfaces in the input expression are lexically analyzed (see *lexical lookup*). The lexical analysis of function words is special, however, in that their core values are variables, e.g. n_1. During the **LA-hear** derivation, the variable of a function word is replaced by the associated content word.

Consider the time-linear derivation of the little dog in 5.1. In combination step 1, the adnominal little is copied into the mdr slot of *the*, and the variable n_1 of *the* is copied into the mdd slot of *little*. In combination step 2, the two occurrences of the variable n_1 in the first two proplets is replaced by the core value of *dog*. Then the third proplet is discarded. In this way, all relevant grammatical relations have been coded into the first two proplets.

Similarly in the time-linear derivation of the complex verb. In combination step 3, the core value dog is copied into the arg slot of the lexical analysis of has, and the variable v_1 is copied into the fnc slot of the *dog* proplet. In combination step 4, the two occurrences of the variable v_1 are replaced by the variable v_2 of the lexical analysis of been. In combination step 5, finally, the occurrences of v_2 are replaced by the core value bark of the last proplet. Then the last proplet is discarded. In the course of this derivation, the semantic contribution of the auxiliaries is coded into the sem slot of the verb (not shown here).[6]

The three proplets derived in this way can now be stored in the data base. Based on the values of the attributes arg and mdr, the proplets may be activated in a VNA navigation. In the speaker-mode, the original surface may be reconstructed as shown in the following derivation:

5.2 SCHEMATIC PRODUCTION OF THE The little dog has been barking.

	activated sequence	*realization*
i		
	V	
i.1	d	d
	V N	
i.2	d an	d an
	V N A	
i.3	d n an	d an n
	V N A	
i.4	ax d n an	d an n ax
	V N A	
i.5	ax ax d n an	d an n ax ax
	V N A	
i.6	ax ax nv d n an	d an n ax ax nv
	V N A	
i.7	ax ax nv p d n an	d an n ax ax nv p
	V N A	

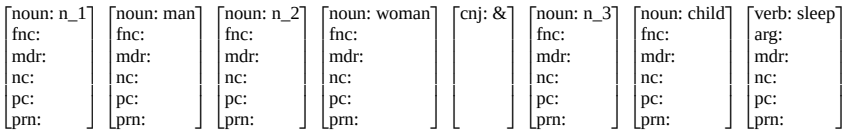
[6] For a more detailed analysis of the major constructions of English see Hausser 2004.

Here, d, an, n, ax, nv, and p stand for determiner, adnominal, auxiliary, non-finite verb, and punctuation (full stop), respectively.

6 Coordination

The most important construction of natural language besides valency in particular and functor-argument structure in general is coordination. For Database Semantics, this presents the task of integrating the treatment of valency and coordination in a unified, functional system. The solution is illustrated below using a simple example of a nominal conjunction with the function word (junctive) and, serving as the subject:

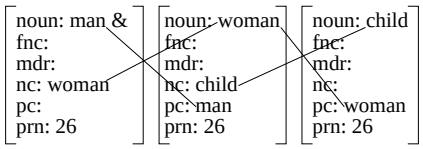
6.1 LEXICALIZATION OF The man, the woman, and the child sleep



The task of connecting these isolated proplets in a time-linear derivation (similar to 5.1) raises two basic questions.

The first is how to build the grammatical relations between the conjuncts. In other words: what is the 'connexion' between the elements of a conjunction in DBS? Consider the following solution:

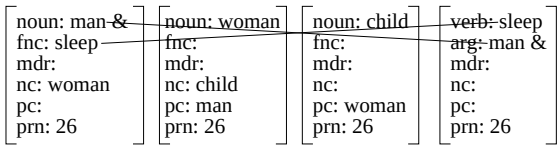
6.2 RELATIONS WITHIN A NOMINAL CONJUNCTION



Each conjunct specifies its predecessor in the pc (previous conjunct) and its successor in the nc (next conjunct) attribute. These attributes receive their values by downward copying in the hearer-mode, and are used for upward retrieval during conceptualization. The kind of conjunction, for example and versus or, is indicated after the concept value of the first conjunct (here man &). This treatment of coordination is strictly surface compositional and time-linear, in contrast to transformational systems such as Hellwig 2003.

The second question is how to integrate such a conjunction into the valency or functor-argument structure of a proposition. Consider the following solution:

6.3 RELATING A NOMINAL CONJUNCTION TO THE VALENCY STRUCTURE



This analysis specifies the grammatical relations of a conjunction in a way which is as complete as necessary and as parsimonious as possible: only the first conjunct *man* specifies the verb in its fnc-slot, and the verb specifies only the first conjunct it its arg-slot. For retrieval, this has the following consequence.

When searching for sleeping child, for example, the *child* proplet in question merely indicates that it is part of a conjunction (pc: woman); in order to determine the associated verb, **LA-think** has to navigate to the first element of the conjunction, i.e. *man*, and check whether or not its fnc value is sleep. The verb proplet *sleep* also indicates that its argument is a conjunction (arg: man &). Therefore, the search for a non-initial conjunct is attempted only if the proplet belongs to a proposition which actually contains a conjunction.

The (re)production of the input sentence is based on a VNNN sequence.[7] The following derivation uses the new surface variable cn, for conjunction:

6.4 PRODUCTION OF The man the woman and the child slept.

	activated sequence				*realization*
i.					
	V				
i.1		d			d
	V	N			
i.2		d nn			d nn
	V	N			
i.3		d nn	d		d nn d
	V	N	N		
i.4		d nn	d nn		d nn d nn
	V	N	N		
i.5		d nn	d nn		d nn d nn
	V	N	N		
i.6		d nn cn	d nn	d	d nn d nn cn d
	V	N	N	N	
i.7		d nn cn	d nn	d nn	d nn d nn cn d nn
	V	N	N	N	
i.8	fv	d nn cn	d nn	d nn	d nn d nn cn d nn fv
	V	N	N	N	
i.8	fv p	d nn cn	d nn	d nn	d nn d nn cn d nn fv p
	V	N	N	N	

The conjunction is lexicalized from the first conjunct in line i.6. Due to this late realization, it appears between the penultimate and ultimate conjunct.

A second kind of intrapropositional conjunction are verbal conjunctions, as in John bought, cooked, and ate the pizza. The relations within a verbal conjunction are similar to those in a nominal conjunction. This approach works also for the combination

[7] Note that VNNN can represent a three-place proposition like John gave Mary a flower or a one-place proposition with a nominal conjunction like 6.3. The notation may be disambiguated by means of subscripts.

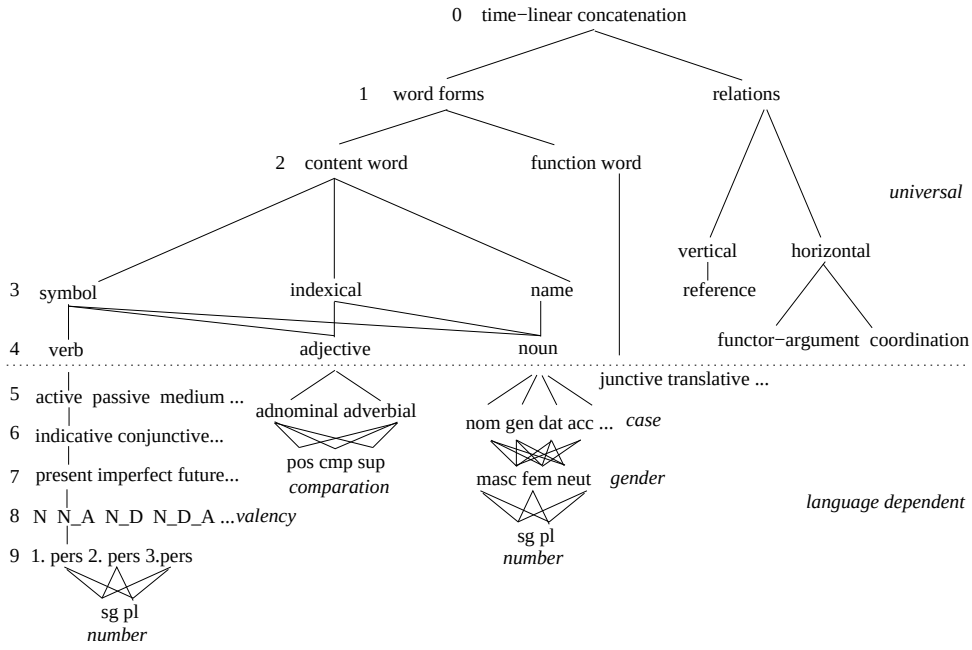
of conjunctions as in the man, the woman, and the child bought, cooked, and ate the steaks, the potatoes, and the broccoli. The functor-argument structure of this example is that of the man& bought& the steaks&.

7 Conclusion

It has been shown in which way the notions of Dependency Grammar based on Valency Theory, such as valency carrier (actant, functor) and valency filler (complement, argument), adjunct (modifier), mot plein (content word), mot vide (function word, translative, junctive), connexion (grammatical relation), etc., have counterparts in Database Semantics. At the same time it became apparent that the realization of these notions and their location in the overall theories is different in the two approaches. This is mainly because Dependency Grammar and Valency Theory are sign-oriented while DBS is agent-oriented.

To further clarify the notions and distinctions in Database Semantics as compared to Dependency Grammar and Valency Theory, consider the following hierarchy:

7.1 HIERARCHY OF NOTIONS AND DISTINCTIONS IN DATABASE SEMANTICS



At the root of the tree there is the time-linear concatenation (level 0) of word forms which are in relations to each other (level 1). This most basic structural property of natural language is realized by the time-linear algorithm of LA-grammar (TCS'92) in its variants of **LA-hear, LA-think**, and **LA-speak** (AIJ'01).

The word forms are divided into content words and function words (level 2). The content words are divided into the three basic kinds of signs, namely symbol, indexical,

and name (level 3). In the branch of relations, the kinds of signs serve the 'vertical' relation of reference, implemented as a matching between the levels of language and context.

The kinds of signs (level 3) are correlated with the parts of speech (level 4). Symbols can be verbs, adjectives, or nouns, indexicals can be adjectives or nouns, and names can be nouns only. This is shown graphically by the lines relating the kinds of signs and the parts of speech. In the branch of relations, the parts of speech serve the 'horizontal' relations of functor-argument structure and coordination.[8]

The structures shown above the dotted line separating level 4 and 5 are universal: all natural languages are based on a time-linear concatenation of word forms, the distinction between content and function words, the three kinds kinds of signs, the three parts of speech, the vertical relation of reference, and the horizontal relations of functor-argument structure and coordination. In Database Semantics, these structures are realized in the form of an artificial agent with interfaces for recognition and action at the context and the language level, the data structure of a word bank (database) containing proplets, and the algorithm of LA-grammar for reading content into and out of the database.

The structures shown below the dotted line are language-dependent. For the verb forms of the Indo-European languages, for example, this holds for the genus, modus, and tempus verbi (levels 5, 6, and 7), the valency structure of the verbs (level 8), as well as the person and number distinction (level 9). For the adjectives, it holds for the distinction between adnominal and adverbial use and for synthetic comparation. For the nouns, it holds for the different case systems, and the number and gender distinctions (which are missing, for example, in Korean). Language-dependent is also whether the coding of grammatical relations and distinctions is handled analytically by means of functions words (e.g. junctives, translatives) or synthetically in terms of morphology. In Database Semantics, these aspects are treated by language-dependent LA-grammars with a suitable lexicon, restrictions on variables for handling agreement, and a rule system for handling word order.

Bibliography

Ágel, V. (2000) *Valenztheorie*. Tübingen: Gunter Narr (Narr Studienbücher).

Greenberg, J. (1963) "Some Universals of Grammar with Particular Reference to the Order of Meaningful Elements," in J. Greenberg (ed.) *Universals of Language*, MIT Press, Cambridge, Mass.

Hausser, R. (1992) "Complexity in Left-Associative Grammar," *Theoretical Computer Science*, Vol. 106.2:283-308, Dordrecht: Elsevier. (TCS'92)

[8] Treating coordination as a bona fi de grammatical relation like functor-argument structure, handled in terms of the attributes and values of proplets, is in contrast to Lobin 1993: "The best way of dealing with coordination is not to deal with it at all, but 'process it away immediately'."

Hausser, R. (1999) *Foundations of Computational Linguistics, Human-Computer Communication in Natural Language*. 2nd Edition 2001, pp. 578, Berlin, New York: Springer-Verlag. (FoCL'99)

Hausser, R. (2001) "Database Semantics for Natural Language," *Artificial Intelligence*, Vol. 130.1:27–74, Dordrecht: Elsevier. (AIJ'01)

Hausser, R. (2004) "The major constructions of English", submitted. Available at http://www.linguistik.uni-erlangen.de/de_contents/publications.php.

Hellwig, P. (2003) "Dependency Unification Grammar." In: *Dependency and Valeny. An International Handbook of Contemporary Research.* Edited by V. Ágel, L.M. Eichinger, H.-W. Eroms, P. Hellwig, H.-J. Heringer, H. Lobin. Mouton.

Herbst, T. (1999) "English Valency Structures - A first sketch." *Erfurt Electronic Studies in English* (EESE) 6/99.

Herbst T., D. Heath, I. F. Roe, & D. Götz (2004) *A Valency Dictionary of English: A Corpus-Based Anaysis of the Complementation Patterns of English Verbs, Nouns and Adjectives*. Berlin/New York: Mouton de Gruyter.

Lobin, H. (1993) "Linguistic Perception and Syntactic Structure." In: *Functional Description of Language.* Edited by E. Hajicova. Charles University, Prague.

Lobin, H. (1993) *Koordinations-Syntax Als Prozedurales Phaenomen*. Tübingen: Gunter Narr.

MacWhinney, B. (1987) "The Competition Model." In B. MacWhinney (Ed.), *Mechanisms of language acquisition* (pp. 249-308). Hillsdale, NJ: Lawrence Erlbaum.

MacWhinney, B. (2004) "A multiple process solution to the logical problem of language acquisition." *Journal of Child Language*, 31, 883-914.

MacWhinney, B. (2005) "Item-based constructions and the logical problem." *ACL* 2005, 46-54.

Tesnière, L. (1959) *Éléments de syntaxe structurale*, Editions Klincksieck, Paris.