Database Semantics:
Natural Language Interpretation, Inferencing, and Production

Roland Hausser

Abstract

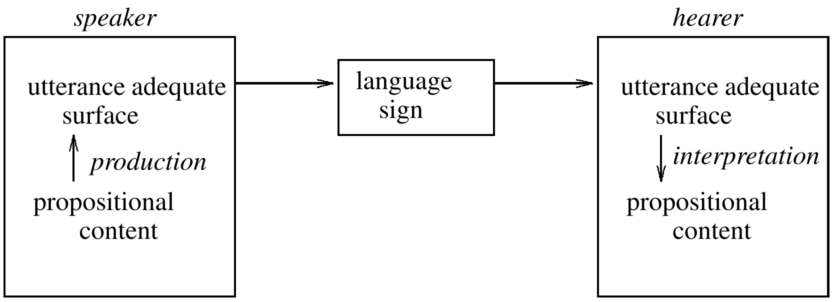
This paper describes a new data structure suitable for a time-linear reading in and out of natural language meanings. It represents propositions as sets of conindexed feature structures, called proplets. Each proplet specifies its intra- and extrapropositional continuation proplets. The navigation through these continuations is the common basis for interpretation, inferencing, and production.

1 The mechanism of natural communication

Natural language communication between a speaker and a hearer may be viewed as a transfer of information from one database to another. Thereby, production involves the mapping of propositional contents into utterance adequate surfaces while interpretation involves the mapping of utterance adequate surfaces into propositional contents.

Utterance adequate surfaces contain indexicals such as pronominal, temporal, and spatial expressions relating to the utterance's location of origin, as well as language specific properties such as the word forms, word order, agreement, etc. The corresponding propositional content, on the other hand, is utterance independent and universal.

1.1 CORRELATION OF INTERPRETATION AND PRODUCTION



An act of communication is successful if the propositional content coded by the speaker into the natural surface is reconstructed by the hearer equivalently.

2 Representing propositions as sets of proplets

The structure of the propositional content must be suitable for a *time-linear* reading of natural language into and out of the database. As a simple example of a propositional content consider The field contains a triangle and the field contains a square. The elementary propositions, concatenated by and, each consist of the relation contain between the subject field and an object.

The following datastructure codes this information in terms of co-indexed proplets. A proplet is a basic element of a proposition, defined as a feature structure.[1]

2.1 CONCATENATED PROPOSITIONS IN A WORD BANK

TYPES	PROPLETS	
[concept: contain role: functor]	[verb: contain ARG:field triangle prn: 23 epr: 23 und 24]	[verb: contain ARG:field square prn: 24 epr: 23 und 24]
[concept: field role: argument]	[np: field FUNC: contain prn: 23 id: 7]	[np: field FUNC: contain prn: 24 id:7]
[concept: square role: argument]	[np: square FUNC: contain prn: 24 id: 9]	
[concept: triangle role: argument]	[np: triangle FUNC: contain prn: 23 id: 8]	

The feature structures representing the types are each used only once and ordered alphabetically. The tokens, called proplets, are ordered behind their types. The horizontal lines, consisting of one type and a sequence of associated proplets, are called *token lines*.

A data structure like 2.1 is called a word bank. Its token lines correspond to a *network database*, which defines a 1:n relation between two kinds of records, the owner records and the member records. In a word bank, the types function as *owner records* and the associated proplets as *member records*.[2]

[1]The treatment of modifiers is omitted for reasons of space.

[2]In the initial phase of evolving a word bank, we are currently using a commercial industrial grade relational database capable of storing and accessing millions of proplets. This is possible because (i) a network database can be simulated within a relational database, and (ii) the definition of possible continuations within the proplets is merely an interpretation of the database structure.

An elementary proposition is defined as a set of proplets with the same proposition number prn. For example, the proplets with the prn 23 in 2.1 constitute one proposition, while the proplets with the prn 24 constitute the other.

The functor-argument structure of an elementary proposition is coded into the intrapropositional continuation features of its proplets. Thereby, nominal proplets specify the related functor and verbal proplets specify the related arguments. For example, the proplet with the name contain and the proposition number 24 in 2.1 has the continuation features [ARG: field square]. Correspondingly, the nominal proplets with the name field and the proposition numbers 23 and 24 each contain the continuation feature [FUNC: contain].

In this way, the grammatical structure of the associated proposition may be reconstructed from any proplet. For example, the second proplet in the second token line of 2.1 specifies that it is (i) of the type field, (ii) has the role of an argument, (iii) belongs to proposition 24, and (iv) that the associated functor is a proplet with the name contain and the prn 24. With this information, the second proplet in the first token line may be found in the database. The proplet confirms (a) that it has the role of a functor and (b) that one of its arguments is of the type field. Furthermore, it provides (c) another continuation, namely the argument square.

The concatenation of elementary propositions is defined in terms of extrapropositional continuation features of the proplets. In nominal proplets, this is the identity number id, which specifies coreference or non-coreference with other nominal proplets. For example, that a triangle and a square are contained in the *same* field is expressed in 2.1 by the field proplets having the same id value (here 7).

In verbal proplets, extrapropositional continuations are specified by their epr feature. For example, the epr feature of the second contain proplet specifies that the preceding proposition has the prn number 23 and that the conjunction of this concatenation is and. When proposition 24 has been traversed intrapropositionally, the epr feature of its verb may be used to navigate to proposition 23.

3 Time-linear reading in of propositional contents

A word bank goes beyond a classical network database only insofar as it defines *possible continuations* within its record-based structure. These form the basis for a kind of operation which conventional databases do not provide, namely a navigation through the database. One kind of navigation is the time-linear reading in of propositional contents during interpretation.

3.1 SYNTACTICO-SEMANTIC ANALYSIS OF field contains triangle

combination step 1:

syntax: **field** (N) + **contains** (N A V) ⟹ **field contains** (A V)

semantics:

$$\begin{bmatrix}\text{np:} & \textbf{field}\\ \text{FUNC:} & \\ \text{prn: 23} & \end{bmatrix} \begin{bmatrix}\text{verb:} & \textbf{contain}\\ \text{ARG:} & \\ \text{prn: 23} & \end{bmatrix} \Longrightarrow \begin{bmatrix}\text{np:} & \textbf{field}\\ \text{FUNC:} & \textbf{contain}\\ \text{prn: 23} & \end{bmatrix} \begin{bmatrix}\text{verb:} & \textbf{contain}\\ \text{ARG:} & \textbf{field}\\ \text{prn: 23} & \end{bmatrix}$$

combination step 2:

syntax: **field contains** (A V) + **triangle** (A)

semantics:

$$\begin{bmatrix}\text{np:} & \textbf{field}\\ \text{FUNC:} & \textbf{contain}\\ \text{prn: 23} & \end{bmatrix} \begin{bmatrix}\text{verb:} & \textbf{contain}\\ \text{ARG:} & \textbf{field}\\ \text{prn: 23} & \end{bmatrix} \begin{bmatrix}\text{np:} & \textbf{triangle}\\ \text{FUNC:} & \\ \text{prn: 23} & \end{bmatrix}$$

syntax: ⟹ **field contains triangle** (V)

semantics: ⟹

$$\begin{bmatrix}\text{np:} & \textbf{field}\\ \text{FUNC:} & \textbf{contain}\\ \text{prn: 23} & \end{bmatrix} \begin{bmatrix}\text{verb:} & \textbf{contain}\\ \text{ARG:} & \textbf{field triangle}\\ \text{prn: 23} & \end{bmatrix} \begin{bmatrix}\text{np:} & \textbf{triangle}\\ \text{FUNC:} & \textbf{contain}\\ \text{prn: 23} & \end{bmatrix}$$

This strictly time-linear analysis consists syntactically in *cancelling* (namely of valency positions by valency fillers) and semantically in *copying* (namely of proplet names into the continuation features of other proplets).

On the syntactic level, the nominative (N) and the accusative (A) valency is cancelled in the first and second combination step, respectively. On the semantic level, the words are assigned feature structures via the lexicon. These contain the name of the proplets, e.g. field, and a continuation attribute with the value NIL (represented here as the empty value). Also, the prn-feature is provided with a suitable proposition number (here 23).

The semantic interpretation of the first syntactic composition consists in copying the proplet name field into the FUNC feature of contain, and the proplet name contain into the ARG feature of field. The interpretation of the second composition provides the new proplet triangle with the FUNC value contain, and the proplet contain with the ARG value triangle.

The result are the three coindexed proplets field, contain, and triangle. They each specify their respective continuation predicates as well as the functor-argument structure of the proposition. The individual proplets are autonomous entities suitable to be added into the word bank in accordance with the structural requirements of a network database.

4 The motor algorithm of navigation

The reading in of propositional content during interpretation (hearer mode) is activated by parsing the incoming language signs. The reading out of propositional content during production (speaker mode), on the other hand, is based on (i) the concatenated propositions in the word bank (rail road system) and (ii) a LA-NA grammar[3] which activates the navigation through the word bank (locomotive).

One form of navigation is the accidental, aimless motion through the contents of a word bank. This free form of navigation is analogous to *free association* in humans, i.e. the giving one's thoughts free rein.

Free association is the best starting point for a general treatment of language production because it requires the automatic realization of arbitrary thought paths as a sequence of natural language expressions. Free association may be overruled by external or internal demands which activate specific navigation patterns appropriate to the situation at hand.

The navigation through coherent propositional contents ensures the coherence of the associated language expressions and provides an essential aspect of cohesion, namely the serialization of the language expressions. In this way, the navigation-based conceptualization minimizes the traditional distinction between content (*what to say*) and form (*how to say it*).

The rules of the motor algorithm have the task of moving the navigation from a given proplet to a possible continuation proplet. A successful rule application is illustrated in 4.1, using the example of an intrapropositional navigation.

4.1 Schema of a LA-NA rule application

$$
\begin{array}{lcccccl}
 & \text{START} & & \text{NEXT} & & \text{NEW START} & \\
\text{rule}_{1+2\Rightarrow 2}: & \begin{bmatrix}\text{m1: } a \\ \text{M2: } b \\ \text{prn: } c\end{bmatrix} & & \begin{bmatrix}\text{m2: } b \\ \text{M1: } x\ a\ y \\ \text{prn: } c\end{bmatrix} & \Longrightarrow & \begin{bmatrix}\text{m2: } b \\ \\ \\ \end{bmatrix} & \text{rule package}_{1+2\Rightarrow 2} \\
 & & & & & & \\
\begin{array}{c}\text{proplets} \\ \text{of the} \\ \text{word bank}\end{array} & \begin{bmatrix}\ldots \\ \text{m1: } c1 \\ \ldots \\ \text{M2: } c2 \\ \ldots \\ \text{prn: } c3 \\ \ldots\end{bmatrix}_1 & + & \begin{bmatrix}\ldots \\ \text{m2: } c2 \\ \ldots \\ \text{M1: } ..c1.. \\ \ldots \\ \text{prn: } c3 \\ \ldots\end{bmatrix}_2 & \Longrightarrow & \begin{bmatrix}\ldots \\ \text{m2: } c2 \\ \ldots \\ \text{M1: } ..c1.. \\ \ldots \\ \text{prn: } c3 \\ \ldots\end{bmatrix}_2 &
\end{array}
$$

First, the start pattern of the LA-NA rule is matched onto the first proplet. This operation is formally based on those attributes which the rule pattern and the start proplet have in common – here m1, M2, and prn. If the start pattern contains attributes not matched by the proplet, the rule is not applicable.

[3] LA-NA stands for 'left-associative navigation.' The algebraic definition of LA-grammar is given in Hausser 1992.

By matching the start pattern onto a word bank proplet, the variables in the pattern are assigned values. For example, the variable *a* is assigned the value *c1*, the variable *b* the value *c2*, and the variable *c* the value *c3*. These assignments hold for the duration of the rule application.

The second step of a LA-NA rule application has the purpose of *finding* the NEXT. This is structurally based on each proplet explicitly specifying its continuation predicates.For example, the START pattern contains the continuation feature [M2: *b*], the variable of which is repeated in the feature [m2: *b*] of the NEXT pattern. Because the variables have been assigned values in the initial step, the proplet matching the NEXT pattern is uniquely determined.

The third step of this particular LA-NA rule application consists in outputting the proplet matching the NEXT as the result (value of the NEW START). The next LA-NA rule takes the proplet of the new START as the sentence start and uses it to look for another next proplet. In this way, a suitable LA-NA syntax may traverse all the intra- and extrapropositional relations of a word bank.

During free navigation, two problems may arise which require a general solution, namely relapse and split. A relapse arises if an LA-navigation continues to traverse the same proplet sequence in a loop. A split arises if a given START proplet is accepted by more than one LA-NA rule in an active rule package.

Relapse and split are avoided by general tracking principles which control free LA-navigation. They are based on counters which indicate for each proplet when it was traversed. Traversal counters give the static nature of a word bank a dynamic aspect insofar as they make the course of the navigation visible as a track.

5 LA-search

Special cases of LA-navigation are (i) LA-*search* and (ii) LA-*inferences*. An LA-search is initiated by a query specifying the desired answer. The two general query types of natural language are called Wh-questions and yes/no-questions.

5.1 Basic types of questions in natural language

Wh-question	**Yes/no-question**
Who entered the restaurant?	Did Peter enter the restaurant?

Formally, the hearer's interpretation of a Wh-question results in a verbal proplet where the value occupied by the Wh-word is represented by the variable σ-1 and the value of prn is represented by σ-2. The hearer's interpretation of a yes/no-question, on the other hand, results in a verbal proplet where only the prn-attribute contains a variable as value.

5.2 Search proplets of the two basic types of queries

Wh-question

$$\begin{bmatrix} \text{verb: } \textit{enter} \\ \text{ARG: } \sigma\text{-1, restaurant} \\ \text{prn: } \sigma\text{-}\mathit{2} \end{bmatrix}$$

Yes/no-question

$$\begin{bmatrix} \text{verb: } \textit{enter} \\ \text{ARG: Peter, restaurant} \\ \text{prn: } \sigma\text{-}\mathit{2} \end{bmatrix}$$

A query is answered by attempting to match the search proplet with the last (most recent) proplet of the corresponding token line (here the line of *enter*).

If the continuation values of this last proplet match the search proplet, the answer has been found and there is no need for further search. Otherwise, the token line is systematically searched, proceeding backwards from the last proplet.

In the case of WH-questions, this search is based on the following LA-grammar.

5.3 LA-Q1 (WH-questions)

ST_S: $\{([a]\{r_1, r_2\})\}$

$$r_1: \begin{bmatrix} \text{verb: } a \\ \neg\text{ARG: } y\ \sigma\ z \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{verb: } a \\ \neg\text{ARG: } y\ \sigma\ z \\ \text{prn: } m-1 \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{verb: } a \\ \neg\text{ARG: } y\ \sigma\ z \\ \text{prn: } m-1 \end{bmatrix} \{r_1\ r_2\}$$

$$r_2: \begin{bmatrix} \text{verb: } a \\ \neg\text{ARG: } y\ \sigma\ z \\ \text{prn: } m \end{bmatrix} \begin{bmatrix} \text{verb: } a \\ \text{ARG: } y\ \sigma\ z \\ \text{prn: } m-1 \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{verb: } a \\ \text{ARG: } y\ \sigma\ z \\ \text{prn: } m-1 \end{bmatrix} \{r_3\}$$

$$r_3: \begin{bmatrix} \text{verb: } a \\ \text{ARG: } y\ \sigma\ z \\ \text{prn: } n \end{bmatrix} \begin{bmatrix} \text{np: } \sigma \\ \text{VERB: } a \\ \text{prn: } n \end{bmatrix} \Longrightarrow \begin{bmatrix} \text{np: } \sigma \\ \text{VERB: } a \\ \text{prn: } n \end{bmatrix} \{\ \}$$

ST_F: $\{([\text{np: } \sigma]\ rp_3)\}$

The proposition numbers in the patterns of LA-Q1 follow the convention that $m-1$ (m minus one) stands for the proposition number of the proplet immediately preceding the current proplet in the token line.

The first two LA-Q1 rules apply if their START-pattern does *not* match the continuation predicates of the input proplet. If the next proplet does likewise not match, then the rule r_1 fires and the search is continued with the next proplet as the new START. If the next proplet matches, on the other hand, rule r_2 fires and the variable σ-1 is bound to the value searched for. Finally, rule r_3 navigates to the proplet of the queried continuation value and returns this proplet as the desired answer.

6 LA-inference

LA-navigation and LA-search have in common that they do not change the propositions contained in a word bank. LA-inference, on the other hand, has the task of deriving new propositions from the information stored, which may result in a modification of the word bank content.

Inferences are needed for many tasks. For example, the linguistic realization of a navigation (production) requires that temporal and modal forms be correctly inferred by the speaker and coded into the language surface. The hearer, in turn, must infer the correct temporal and modal parameter values from those surfaces. Furthermore, the coreference between nominal proplets must be coded into the language surfaces by the speaker using pronouns or definite descriptions, which in turn must be decoded by the hearer. Also, in the answering of questions a hierarchy of hyper- and hyponyms must be used to infer instantiations which are not contained directly in the word bank, etc.

Inferences of this kind have been studied in detail in the logical systems from antiquity to contemporary A.I.[4] This raises the question of whether and how the inferences of the classical as well as the modern systems should be recreated within the formalism of a word bank. As a simple example, consider some classical inferences of propositional calculus.

6.1 Inference schemata of propositional calculus

1. $\dfrac{A, B}{\vdash\ A \& B}$ 2. $\dfrac{A \vee B, \neg A}{\vdash B}$ 3. $\dfrac{A \rightarrow B, A}{\vdash\ B}$ 4. $\dfrac{A \rightarrow B, \neg B}{\vdash\ \neg A}$

5. $\dfrac{A \& B}{\vdash\ A}$ 6. $\dfrac{A}{\vdash\ A \vee B}$ 7. $\dfrac{\neg A}{\vdash\ A \rightarrow B}$ 8. $\dfrac{\neg\neg A}{\vdash\ A}$

In propositional logic, the first inference is called conjunction. Conjunction means that the truth of two arbitrary propositions *A* and of *B* implies the truth of the complex proposition *A&B*.

If this inference is transferred into database semantics, it amounts to an operation which establishes new extrapropositional relations based on the conjunction and. This operation may be realized as the following LA-grammar rule.

6.2 LA-rule for the propositional inference of conjunction

$$\text{inf1:} \begin{bmatrix}\text{verb: } a\\ \text{ARG: } c\\ \text{prn: } m\end{bmatrix} \begin{bmatrix}\text{verb: } b\\ \text{ARG: } d\\ \text{prn: } n\end{bmatrix} \Longrightarrow \begin{bmatrix}\text{verb: } a\\ \text{ARG: } c\\ \text{prn: } m\\ \text{epr: } m \textit{ and } n\end{bmatrix} \begin{bmatrix}\text{verb: } b\\ \text{ARG: } d\\ \text{prn: } n\\ \text{epr: } m \textit{ and } n\end{bmatrix}$$

[4] Cf. Bibel 1993.

The rule inf1 produces the new extrapropositional relation [epr: *m and n*] between two arbitrary propositions *m* and *n*, enabling navigation from any proposition to any other proposition asserted in a word bank.

In its logical interpretation, the inference in question expresses a conjunction of truth and is as such intuitively obvious. The word bank interpretation, on the other hand, raises the question of *why* two – up to now unconnected – propositions (data base assertions) should be concatenated with and. Even though such a concatenation would not result in a falsehood, an uncontrolled application of inf1 would destroy the cognitive coherence of a word bank.

Another type of inference models the *is-a*, *is-part-of* and *has-a* hierarchies of classic A.I. In a word bank, these conceptual hierarchies are represented in terms of *absolute* propositions. A natural language correlate of an absolute proposition is, e.g., A dog is an animal.

Absolute propositions express general knowledge, in contrast to the *episodic* propositions considered up to now, which express individual knowledge about specific events and objects, such as Peter crosses the street. Formally, absolute propositions differ from episodic propositions only in that their spatio-temporal origin is unspecified. Other than that absolute propositions form normal contents of a word bank, which can be traversed both intra- and extrapropositionally.

Absolute propositions are an important component of a word bank and serve in part to specify the literal meaning of words. In the token lines of a word bank, the proplets of absolute propositons are positioned between the respective base form (type) and the episodic proplets (cf. Hausser 1996). As an example of an absolute proposition consider 6.3.

6.3 Proplets of an absolute proposition

$$\begin{bmatrix}\text{verb: } \textit{be} \\ \text{ARG: dog, animal} \\ \text{prn: abs327}\end{bmatrix} \quad \begin{bmatrix}\text{np: } \textit{dog} \\ \text{FUNC: be} \\ \text{prn: abs327}\end{bmatrix} \quad \begin{bmatrix}\text{np: } \textit{animal} \\ \text{FUNC: be} \\ \text{prn: abs327}\end{bmatrix}$$

These proplets express the absolute proposition A dog is an animal. This way of formalizing absolute propositions is suitable to express any of the usual conceptual hierarchies in a word bank. At the same time, such absolute propositions serve as the basis for various kinds of inferences.

Assume, for example, that the word bank contains the following proplet as part of the episodic proposition Peter saw a dog.

6.4 Proplet of an episodic proposition

$$\begin{bmatrix}\text{verb: } \textit{see} \\ \text{ARG: Peter, dog} \\ \text{prn: 969}\end{bmatrix}$$

Without further provision, the question Did Peter see an animal? would be answered with no by the word bank. However, given the absolute proposition 6.3

and the following rule of inference inf2, the word bank could infer that Peter saw indeed an animal when he saw the dog and answer the question correctly with yes.

6.5 Inference rule inf2 for absolute propositions

$$\text{inf2:}\quad \begin{bmatrix}\text{verb: } a\\ \text{ARG: } x\ b\ y\\ \text{prn: } n\end{bmatrix} \begin{bmatrix}\text{verb: } be\\ \text{ARG: } b\ c\\ \text{prn: } abs\end{bmatrix} \Longrightarrow \begin{bmatrix}\text{verb: } a\\ \text{ARG: } x\ c\ y\\ \text{prn: } n\end{bmatrix}$$

This rule is applicable to verbal proplets of all absolute propositions with the verb be, i.e., to all the elements of the *is-a* hierarchy. Similar rules may be defined for the other hierarchies.

In our example, the START-pattern of rule inf-2 is matched onto the episodic proplet 6.4, whereby the variable b is assigned the value dog. This enables the rule inf-2 to navigate to the verbal proplet of the absolute proposition 6.5 as the NEXT and to derive the new episodic proposition Peter saw an animal as a variant of the original proposition. Based on this inference, the question Did Peter see an animal? is answered correctly, even though the word bank originally contained only the proposition Peter saw a dog.

References

Bibel, W. (1993) *Wissensrepräsentation und Inferenz*, Vieweg, Braunschweig-Wiesbaden.

Hausser, R. (1989). *Computation of Language*, Symbolic Computation: Artificial Intelligence, Springer-Verlag, Berlin-New York.

Hausser, R. (1992) "Complexity in Left-Associative Grammar," *Theoretical Computer Science*, Vol. 103, Elsevier.

Hausser, R. (1996) "A Database Interpretation of Natural Language," *Korean Journal of Linguistics*, Vol. 21, No. 1,2:29–55.

Hausser, R. (1999). *Foundations of Computational Linguistics*, Springer-Verlag, Berlin-New York.

Russell, S.J. & P. Norvig (1995) *Artificial Intelligence, a modern approach*, Prentice Hall, New Jersey.