Signal Processing 32 (1993) 111-145
Elsevier

Control and explanation in a signal understanding environment

F. Kummert[1], H. Niemann[2], R. Prechtel[2] and G. Sagerer[1]
[1]*Universität Bielefeld, Technische Fakultät, Postfach 8640, W-4800 Bielefeld 1, Germany*
[2]*University of Erlangen-Nürnberg, Informatik 5 (Mustererkennung), Martensstrasse 3, W-8520 Erlangen, Germany*

Received 5 March 1992
Revised 27 May 1992

Abstract. To interpret sensor signals like images, image sequences, or continuous speech the representation and use of task-specific knowledge is necessary. The paper presents a framework for the representation of declarative and procedural knowledge using a suitable definition of a semantic network. Based on that formalism a problem-independent control algorithm for the interpretation of sensor signals is presented. It provides both data-driven and model-driven control structures which can easily be combined to perform any mixed strategy. An explanation facility is available which makes the development of complex knowledge bases easier and increases the acceptance of such a knowledge-based analysis system.

Zusammenfassung. Unerläßliche Voraussetzung für die Interpretation von Sensordaten wie Bilder, Bildfolgen oder kontinuierliche Sprache ist die Darstellung und Nutzung von aufgabenspezifischem Wissen. In diesem Artikel wird ein Formalismus vorgestellt, der deklaratives und prozedurales Wissen in einem semantischen Netzwerk repräsentiert. Auf diesem Formalismus aufbauend ist ein problemunabhängiger Kontrollalgorithmus für die Interpretation von Sensordaten definiert. Er besitzt sowohl datengetriebene als auch erwartungsgesteuerte Kontrollmechanismen, wodurch jede gemischte Strategie erzielt werden kann. Eine Erklärungskomponente unterstützt die Entwicklung und Wartung komplexer Wissensbasen und erhöht die Akzeptanz eines solchen wissensbasierten Analysesystems.

Résumé. L'interprétation de signaux captés comme les images, les séquence d'images ou encore la parole rend nécessaire l'usage d'une représentation de la connaissance spécifique à la tâche. L'article présente un environment de travail pour la représentation de connaissances declaratives et procedurales faisant appel a un réseau sémantique dédié. Un algorithme de contrôle indépendant du problème utilise ce formalisme pour l'interprétation de tels signaux. Il permet le développement de structures de contrôle orientées à la fois données et modèle. Toute combinaison, car celle-ci est facilitée, peut conduire à adopter une stratégie mixte. Une composante explicative est disponible qui facilite le développement de bases de connaissances importantes et qui augmente les perspectives que renferme un tel systéme d'analyse de bases de connaisances.

Keywords. Knowledge representation; semantic networks; interpretation of sensor signals; problem-independent control; explanation facility.

1. Introduction

The task of a system for signal understanding is to compute a symbolic description $\mathscr{B}$ of an input signal f which
- optimally fits to the input signal,
- is maximally consistent with internally represented task-specific knowledge,
- and contains the information necessary for further processing steps within the task-domain.

Typical examples of signals are speech, images, or image sequences. At least parts of the relevant knowledge have to be represented explicitly within

Correspondence to: Dr. F. Kummert, Universität Bielefeld, Technische Fakultät, Angewandte Informatik, Postfach 8640, W-4800 Bielefeld 1, Germany. e-mail: franz@techfak.uni-bielefeld.de.

the processing system in order to speak of a knowledge-based system.

The main modules of a system for knowledge-based signal understanding are shown in Fig. 1. Four of them are essential for any task; they are the methods for performing an initial segmentation, the representation and utilization of knowledge, the representation of intermediate results of processing, and the control of an understanding process. The desirability of the additional modules for knowledge acquisition, explanation and user interface depends on the intended usage of the system. This article concentrates on an approach to the design and implementation of the modules for knowledge representation and utilization, control and explanation.

Early books and articles on knowledge-based processing are, for example, [5, 12, 19, 20, 24, 27, 28, 40, 58]. Some examples of recent developments are given in [25, 30, 33, 39, 42]. We may view signal understanding as a problem-solving activity. Then it is natural to distinguish various states of problem solving or of processing. From the above early references two viewpoints (and many intermediate ones) can be distinguished:

One viewpoint is that states are represented by the data or results computed by the system. A set of operators, procedures, or transformations is given to the system which allows it to transform one state into another one which hopefully is closer to the goal of processing (that is to a symbolic description). Control then amounts to the selection of a subset of the data and a transformation to work on this subset. Only that subset of states and state transitions relevant to interpret an input signal is generated. The approach described here is of this type.

The other viewpoint is that states represent possible partial interpretations of the signal. Knowledge constrains the set of states and of possible state transitions. Control then amounts to a search of a path leading from the initial state to a goal state. States and transitions are precomputed and represented in a network.

Of central importance to a knowledge-based system is, of course, the formalism employed for knowledge representation. In general, knowledge can be represented by formal logic [38, 57], fuzzy logic [2, 7, 8, 60], frames and semantic networks [10, 14, 17, 22, 30, 59], production systems or rule-based systems [3, 24, 26, 52] or (artificial) neural networks [13, 41]. It would be beyond the scope of this contribution to discuss the relative merits of the different approaches. This paper is based on knowledge representation by semantic networks. A motivation for this approach is given in [34, 45] in detail.

Briefly, the approach to knowledge-based signal understanding is as follows. At first a mainly data-driven phase of processing is carried out to compute an initial segmentation $\mathscr{A}$ of the input signal f. This way the signal is decomposed into segmentation objects O using no task-specific knowledge. A segmentation object is a recursive data structure,

$$O=(D\colon T_O,\ (A\colon (T_A,\ \mathscr{R}\cup V_T))^*,\ (P\colon O)^*,\ (S(A_O, A_P, A_K)\colon \mathscr{R})^*,\ G\colon \mathscr{R}^n) \qquad (1)$$

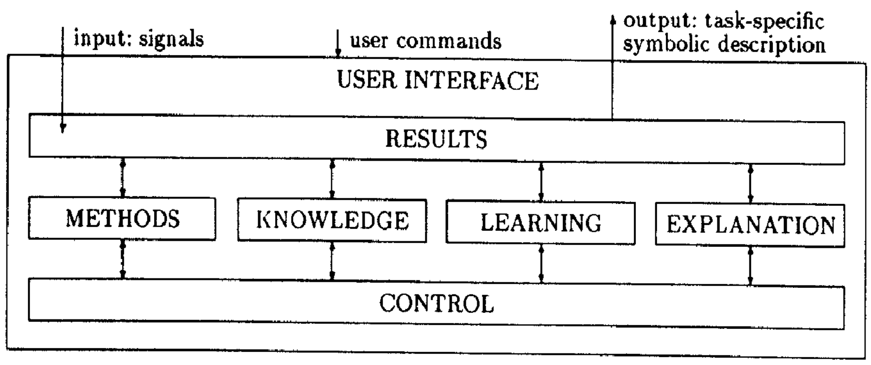


Fig. 1. The main modules of a system for knowledge-based signal understanding.

having attributes A, parts P, structural relations S and a judgment G. Attributes are defined by their type T_A, for example, length (of a line) or color (of a region), and the associated value which may be a real number $\mathscr{R}$ or a symbol from some alphabet V_T; parts are themselves segmentation objects O, for example, a 'pair of parallel lines' has as its parts two 'straight lines'; structural relations occur between attributes A_O of the segmentation object or attributes A_P of its parts, for example, the relation between the two straight lines of a parallel line pair is the constant distance which is treated as a fuzzy relation whose degree of fulfilment is measured by a real number $\mathscr{R}$; finally, the judgment in general is a vector measuring, for example, goodness and priority by real numbers $\mathscr{R}$.

An initial segmentation is a network of segmentation objects,

$$\mathscr{A} = \langle O \rangle. \tag{2}$$

Initial segmentation is followed by a phase of mainly model-driven processing. Knowledge is represented in a model $\mathscr{M}$ of the task-domain; it consists of concepts C of a semantic network. Hence, a model is a network of concepts,

$$\mathscr{M} = \langle C \rangle. \tag{3}$$

Results of processing are represented as instances $I(C_i)$ of concepts C_i. We assume that the goal of processing is itself represented by a concept, the so called goal concept C_g. Every concept has an associated judgment G which allows one to compute the judgment (or goodness, score) of an instance. Hence we view knowledge-based signal understanding as the problem to compute an optimal instance $I^*(C_g)$ of a goal concept. Now we can define the above mentioned symbolic description $\mathscr{B}$ by

$$\mathscr{B}(f) = I^*(C_g),$$
$$I^*(C_g) = \operatorname*{argmax}_{\{I(C_g)\}} \{G(I(C_g)) \mid \mathscr{M}, \mathscr{A}\}. \tag{4}$$

An optimal instance can be computed by an adaptation of the well-known A^*-graph-search algorithm to semantic networks as shown in Section 3. Details of this approach are given below. In summary, we treat knowledge-based signal understanding as an optimization problem, which seems appropriate and necessary due to the noisy nature of signals.

The paper gives in the next section an overview of a special version of a semantic network developed to meet the needs of signal understanding. Since this material is available in detail elsewhere, for example in [30, 42, 45], the discussion will be short. The problem of control is treated in detail in Section 3. Additionally to [34], the top-down and bottom-up constraint propagation and the treatment of specializations and optional links are described. The explanation of system resources, that is the declarative and procedural knowledge, as well as of system results computed during an understanding process is presented in Section 4. Two applications in Section 5 show that the presented approach is able to handle different task-domains in an efficient manner. Finally, in Section 6 we give some conclusions and an outlook on further work.

2. The semantic network language

In this section the particular semantic network language ERNEST is presented. There is a clear distinction of the syntax, semantics and pragmatics of the network (not of the task-domain). By 'syntax' we mean the available data structures and the necessary restrictions without regard to their relation to a particular meaning of these structures. The meaning of the data structures, in particular the meaning of nodes, links and substructures, is the 'semantics' of the network. It is important to note that a semantic network in an image or speech understanding system not only has to represent some declarative and procedural knowledge, but also has to provide the basis for its utilization for knowledge-based signal understanding. This aspect is described as the 'pragmatics' of the network in the second subsection.

2.1. The syntax and semantics of the network

Contrary to other approaches, in our definition of the network there exist only three different types of nodes and three different types of links between concepts. They have a well defined semantics and we think that these structures are adequate to represent the knowledge of different signal understanding tasks. In [34, 45] a detailed discussion of the epistemological adequacy of the network formalism can be found. In the following, the example of a jeep is used to illustrate the realization of a network and the meaning of the different structures. Figure 2(a, b) show a rough model of two jeeps, and Fig. 2(c) shows the related network.

Types of nodes

In a knowledge-based system, a basic requirement is the ability to represent classes of objects, events, or abstract conceptions. This is done in ERNEST (and in other approaches to semantic networks) by the node type **concept**. For our example, this results in concepts like Jeep, Tyre, Bodywork, and so on, which describe the common properties of the related objects. Furthermore, to interpret segmented images concepts for the corresponding geometrical terms exist, i.e. Circle, Polygon.

In the context of image or speech understanding an important step is the interpretation of the sensor signal in terms modeled in the knowledge base. That means, one connects certain areas of the signal with concepts of the knowledge base. For that, the second node type, called **instance**, is introduced which represents an extension of a concept found in the sensor data. The instance is a copy of the related concept except that the general description is substituted by concrete values calculated from the signal data.

In an intermediate state of processing it may occur that instances to some concepts cannot be computed because certain prerequisites are missing. Nevertheless, the available information can be used to constrain an uninstantiated concept. This is done by the node type **modified concept** which represents modifications of a concept due to intermediate results of the analysis. For a clear distinction between a term and the related model in the network, the following convention is used: the term *xyz* is represented by the concept *Xyz*. An instance to *Xyz* is denoted by *I*(*Xyz*), a modified concept by *M*(*Xyz*).

Types of links

Like in all approaches to semantic networks there exists a link type **specialization** which connects a concept with a more general concept (i.e. Car $\xrightarrow{\text{spec}}$ Jeep). Closely related to that link is an inheritance mechanism by which a special concept inherits all properties of its general ones, unless they are explicitly modified.

Another well-known link type is **part** which decomposes a concept in its natural components (i.e. Car $\xrightarrow{\text{part}}$ Tyre). However, in image or speech understanding, it often occurs that a certain concept is only defined in the context of another one. For example, if you want to find a spare tyre in an image it can be only identified as a spare tyre in the context of a related vehicle. Contrarily, an ordinary tyre can be recognized without any context as the definition of that term is independent of relationships to other ones. However, the term front tyre is context-dependent as this property can be only determined by an appropriate context. To model these facts a part can be marked as **context-dependent** and vice verse a context can be explicitly inserted in a concept. That means, Spare_tyre is a context-dependent part ($\xrightarrow{\text{cdpart}}$) of Jeep and in Spare_tyre the concept Jeep is inserted as a possible context. Dependent on the choice of the created concepts, there are various possibilities to design a network. For example, if you want to model the objects front tyre and rear tyre explicitly as concepts Front_tyre and Rear_tyre are context-dependent parts of Car. In the other case, where only the concept Tyre exists, two context-independent links from Car to Tyre are introduced. To express the above relation for every link a **functional role** is

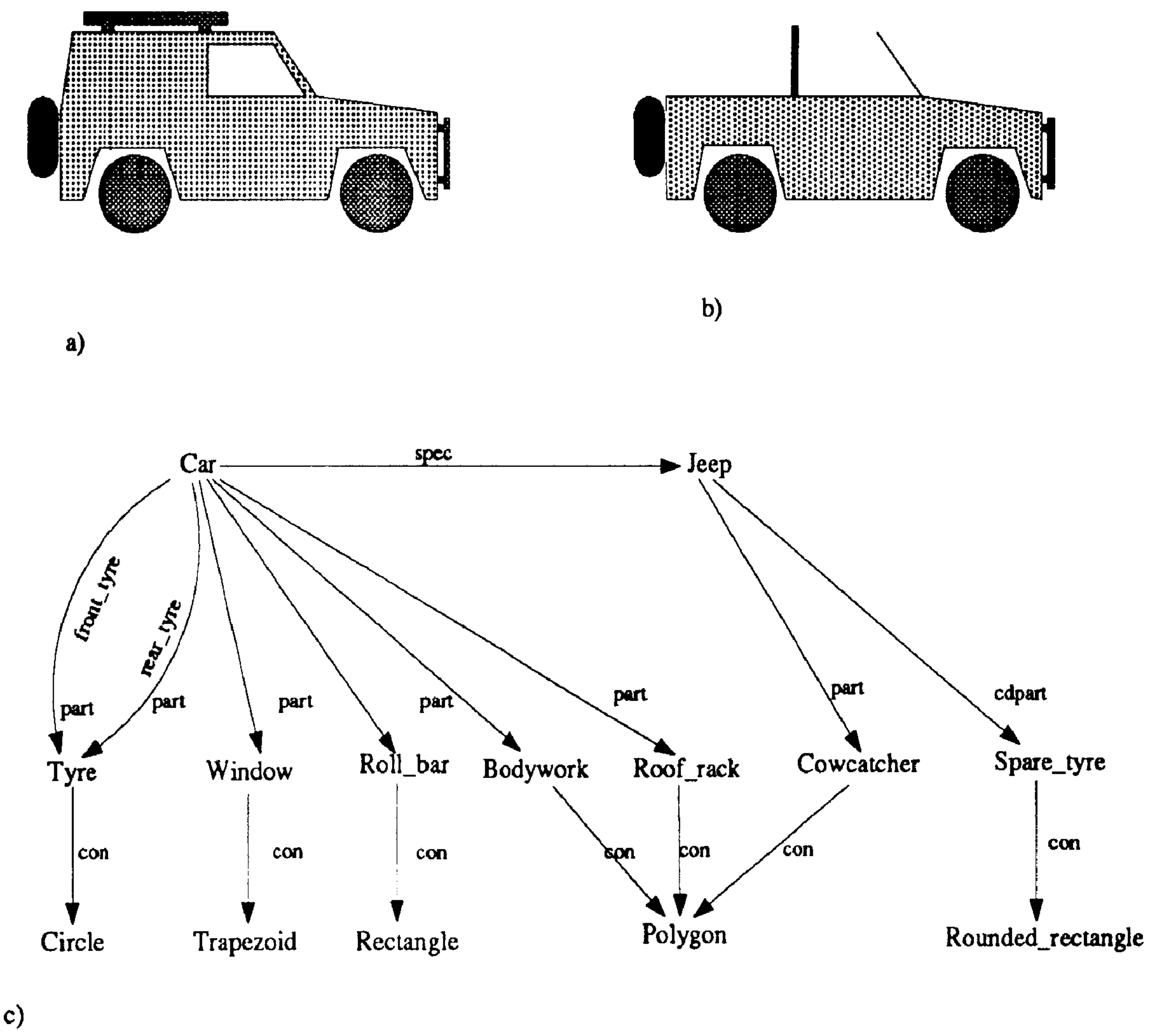


Fig. 2. A simple graphic representation of two jeeps and the related network model.

defined (see Fig. 2(c)). Generally, for every link such a role exists. To guarantee the clarity these roles are omitted for the other links in Fig. 2(c).

In order to motivate the third link type, the description of aggregation in [23] is reported: "for example, the parts of John Smith, viewed as a physical object, are his head, arm, etc. When viewing as a social object, they are its address, social insurance number, etc." Two conceptional systems are distinguished in this example. A concept modeling a person has different parts within each of these systems. Parts in the social system are social conceptions, parts in the physical system are physical conceptions. In complex applications, more than one such conceptional system will occur, i.e., in image understanding, lines, geometry, named objects, or motions. Relationships between concepts belonging to different conceptional systems are only established by the link type **concrete**. Therefore, part and specialization are restricted in the way that they are only allowed inside the same conceptional system. For example, the concepts Tyre and Circle represent terms of different conceptional systems because tyre belongs to 'named object', while circle belongs to 'geometry'. According to the fact that circle is more concrete to the signal than tyre, the following link Tyre $\xrightarrow{\text{con}}$ Circle is established.

Besides the functional role, the goal of the link, and the slot context-dependent for every part-link and concrete-link a **judgment function** can be defined. It tests the restrictions which are valid for a concept if it is the goal of that link. Furthermore, to support a constraint propagation an **inverse judgment function** is possible. It propagates the restrictions which are tested by the related judgment function. For example, if for the link Jeep $\xrightarrow{\text{part}}$ Tyre a function tests the size of the tyre, then the inverse function can propagate the admissible values to Tyre. By that process, the analysis of an image can be focused only on appropriate hypotheses (see Section 3).

Modality sets

In the definition of a concept, there may be parts and/or concretes (including inherited ones) which are obligatory and others which are optional. A set of obligatory parts and concretes together with the associated set of optional parts and concretes is called a **modality set**. In order to increase the compactness of a knowledge base, we allow that a concept is defined by several modality sets. Each individual modality set is sufficient to compute an instance. Besides the items 'obligatory' and 'optional', inherent links can be defined for such a set. Inherent parts and concretes are those which can be inferred from the instantiation of a concept, but which are not manifested in the sensor signal. For example, when seeing a jeep (or after computing an instance of the concept Jeep), one may usually assume that it has an engine, although this will not be visible under standard viewing conditions. In our example, Jeep has the following two modality sets (see Fig. 3) representing the two jeeps of Fig. 2(a, b). A link in a modality set is referred by the related functional role. Except for Tyre, this is the name of the goal concept in small letters.

Attributes, relations and judgments

Additionally, a concept can be described by **attributes** which represent numerical or symbolic features of a concept. For example, possible attributes for *Car* are *height*, *length* or *speed*. The different attributes of a concept are characterized by a **functional role** which is definite for that concept. For every attribute a **type of value** and a function for the **computation of values** have to be defined. The type reflects the general data type of an attribute and can actually be set to Boolean, Character, Integer, Real, Set, Tree and Record. The computation of an actual value of the attribute can be specified by certain arguments. These arguments can be attributes of the same concepts or can be attributes of concepts which are referred by a link part or concrete. For example, the height of Car can be calculated from the *height* of Bodywork plus the *radius* of Tyre. Besides these obligatory items, a restriction, an inverse function for the computation of a value, and a judgment function can be defined. The **restriction** specifies the allowed or expected values of an attribute. An example is the attribute *speed* of Car which can be restricted to the interval [0; 100] m/h. Similar to the inverse judgment of a link the **inverse computation of a value** supports the constraint propagation. For example, if

- the attribute a is calculated from the attributes b and c via $a = b + c$,
- for a the following restriction is known $\text{restr}_a = [70; 100]$, and
- for c the value 50 is calculated,

then by the inverse computation $b = a - c$ a restriction for b $\text{restr}_b = [20; 50]$ can be propagated. The

modality(1):	Obligatory:	front_tyre, rear_tyre, bodywork, window, spare_tyre
	Optional:	roof_rack, cowcatcher
	Inherent:	-
modality(2):	Obligatory:	front_tyre, rear_tyre, bodywork, spare_tyre
	Optional:	roll_bar, cowcatcher
	Inherent:	-

Fig. 3. Two modality sets of the concept Jeep.

judgment of an attribute reflects the quality of a calculated value with regard to the restriction.

For every concept, **relations** can be specified which define a relationship between different attributes, i.e. *'height_of_car < length_of_car'*. Every relation has a definite **functional role** and a **judgment function**. The value returned by that function measures the degree of fulfillment of the relation. Analogous to the inverse judgment of a link, an **inverse judgment of relation** can be specified which propagates costraints to guarantee the fulfillment of the relation (see the next subsection).

As the results of initial segmentation are often not perfect, an instance to a concept may be more or less erroneous. For that reason, the definition of a concept is completed by a **judgment function** which calculates the correspondence of an area of the sensor signal with the term defined by the related concept. Arguments of this function are the judgments of links, attributes and relations. The scheme to judge an instance is not fixed. Depending on the applications, fuzzy logic, distance measurements or probabilities are used.

So far, the most important structures of the semantic network language are presented. For a detailed description see [34, 45].

2.2. *The pragmatics of the network*

Besides the adequate representation of knowledge, another important aspect of a semantic network is the utilization of this formalism for signal interpretation. The main activity during an analysis process is the computation of instances out of concepts given certain sensor data. Computation of instances only depends on the semantics of the network. It will be shown that six rules are sufficient to define the instantiation process. These rules complete the definition of the network by defining the pragmatics of the formalism in the sense of the procedural semantics [16]. Different from PSN, the rules for generating instances are defined globally for the whole network, without respect to a task-domain. This is possible because the rules only make use of the semantics of the network language, not of the meaning of concepts in a specific network. The six rules are the basis for the problem-independent control algorithm as discussed in the next section.

In the following, the rules will be illustrated by the network in Fig. 2. It is assumed that the goal concept for the instantiation is the concept Jeep with respect to the first modality set (see Fig. 3). The goal of an analysis process is the interpretation of a sensor signal in terms of the knowledge base. This is done by instances which represent these terms with calculated values due to the signal. The inference process is based on the fact that if you have all parts of an object which can be taken apart then you can put it together. In terms of our semantic network, this means: if you have for a concept *A* instances to all obligatory parts and concretes then you can instantiate concept *A*. For example, to instantiate the concept Jeep instances are required to the concepts Tyre, Window, Bodywork and Spare_tyre. As mentioned above, the concept Spare_tyre is only defined in the context of Jeep; that means, for the instantiation of Spare_tyre an instance to Jeep is necessary. Vice versa, a complete Jeep (due to the model) can only be instantiated if Spare_tyre is found in the signal. This conflict is solved in the following way:

1. You create a partial instance to Jeep which is only based on the context-independent obligatory parts Tyre, Window and Bodywork.
2. By that partial instance a context for Spare_tyre is established and an instance can be created.
3. By the instance *I*(Spare_tyre) the partial instance to Jeep can be completed.

This process is expressed by the first two rules for instantiation. Hereby, the first step is described by RULE1 (see Fig. 4).

Because of context-dependent parts and optional links, values do not exist for all arguments of the activated functions. Nevertheless, the knowledge of existing values of arguments and the function itself can be used if the following strategy is applied. The restriction values are also transferred to the functions. The functions themselves decide whether

```
RULE1: Creation of Partial Instances
IF     for a concept A or a modified concept $M_j(A)$ with respect to one obligatory
       set of a modality of A,
       instances exist for all
       • obligatory concretes and
       • obligatory parts which are not context-dependent from A
       AND
       a partial instance exists for a concept which defines a context for A
THEN   build up partial instances $Ip_k(A)$ as follows:
       • create for $Ip_k(A)$ an empty instance,
       • insert the actual modality set,
       • connect $Ip_k(A)$ with those instances referred to by the premise,
       • activate the attached functions for $Ip_k(A)$ in the sequence: judgment of
         links, computation of attributes, judgment of attributes, judgment of
         relations, judgment of the concept A
```

Fig. 4. Rule for the creation of partial instances.

the existing values and the restrictions are sufficient for the estimation of results. In the case of attributes, this estimation is a new tighter restriction. For the other cases, which are all judgments, the estimation must be optimistic. For example, the attribute length of Jeep is the sum of the length of Bodywork and Spare_tyre. By knowing the length a of Bodywork (calculated in I(Bodywork)) and a restriction (interval $[b, c]$) for the length of Spare_tyre (defined in the concept) a new restriction $[a+b, a+c]$ for the length of Ip(Bodywork) can be calculated.

In the situation of Fig. 5 instances for Tyre, Window and Bodywork are already generated. Due to RULE1 a partial instance to Jeep can be created as Jeep has no context and no concretes, and instances for all context-independent parts are available. After that, RULE1 is applicable for Spare_tyre as the context is established and an instance exists for the only concrete. To complete the instantiation process RULE2 (see Fig. 6) is introduced which generates complete instances out of partial ones.

As Spare_tyre has no context-dependent parts, RULE2 is immediately applicable to Ip(Spare_tyre) and after that a complete instance for Jeep can be built up by RULE2. If RULE1 is always applicable to a concept A, then it is called initializing concept because a partial instance can be created only on the basis of segmentation data. Thereby the analysis process is initialized. That means, every concept with no context and only

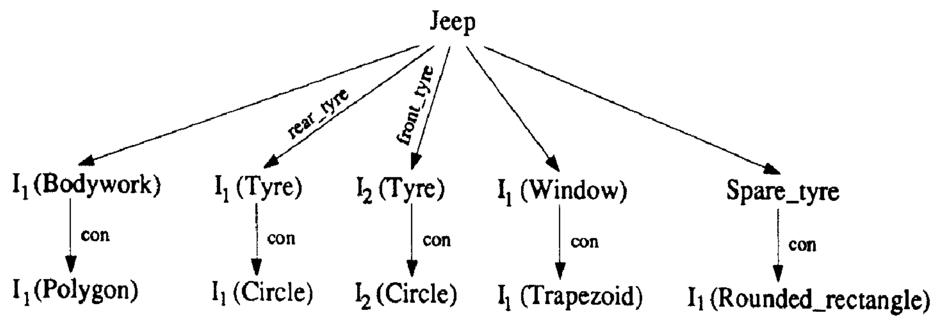


Fig. 5. An example of a state during the analysis.

```
RULE2: Creation of Instances
IF     for a partial instance Ip_j(A)
       instances exist for all context-dependent parts which are obligatory due to
       the modality of Ip_j(A)
THEN   create instances I_k(A) out of Ip_j(A) as follows:
       • create for I_k(A) an empty instance,
       • insert in I_k(A) the modality set of Ip_j(A),
       • connect I_k(A) with those instances referred to by the premise,
       • copy the remaining links of Ip_j(A) to I_k(A)
       • activate the attached functions for I_k(A) as in RULE1
```

Fig. 6. Rule for the creation of instances.

with context-dependent parts can be instantiated at once by RULE1. In the example of Fig. 2 this is the case for the concepts Circle, Rectangle, Trapezoid, Rounded_rectangle and Polygon.

Since RULE1 and RULE2 only consider obligatory parts and concretes, RULE3 in Fig. 7 checks whether there are instances of optional parts or concretes. If this is the case, an instance is extended by these optional components. For example, if there exists an instance for Roof_rack, then *I*(Jeep) can be extended.

If a goal concept for an analysis process is known, recursive application of these three rules results in a search tree for the goal concept according to the modalities of a concept and optional links. By competing instances, generated for a concept, this search tree is additionally expanded. Based on the judgments for instances, the A^*-Algorithm can be used to direct the analysis (see Section 3).

The RULES1-3 are sufficient to define the instantiation of concepts if no modified concepts are allowed. As mentioned above, a modified concept of *A* can be computed if some instances have been computed, but instantiation of *A* is not yet possible. For example, having an instance of Bodywork a modified concept *M*(Jeep) can be calculated. This modified concept represents a model of a jeep where the bodywork and therefore the location in the image is known. This data-driven modification is described by RULE4, see Fig. 8, which provides bottom-up constraint propagation.

The knowledge of *M*(Jeep) may be used to restrict Window, Tyre and Spare_tyre, i.e., concerning the location in the image. By propagating these constraints, the number of competing instances can be reduced. This model-driven creation of modified concepts is expressed by RULE5, see Fig. 9, which provides top-down constraint propagation.

For example, as Tyre can be referred to by other concepts than Jeep (i.e. Truck) the restriction for the attribute radius in Tyre has to take into account all these references. By the model-driven modification of Tyre (due to *M*(Jeep)), a new restriction for the radius of a tyre of jeep can be inserted. This is done by the inverse judgment of the link Jeep $\xrightarrow{\text{part}}$ Tyre by propagating the admissible

```
RULE3: Creation of extended instances
IF     for an instance I_j(A)
       there exist instances which are optional due to the modality set of I_j(A)
THEN   create extended instances I_k(A) out of I_j(A) as described in RULE2
```

Fig. 7. Rule for the creation of extended instances.

RULE4: *Data-driven modification of concepts*

IF for a concept A or a modified concept $M_j(A)$ and a modality set of A new modified concepts or new instances were created for a concept, which is referred to as part, or concrete, or context-of by the concept A

THEN create new modified concepts $M_k(A)$ out of A or $M_j(A)$, respectively, as follows:

- create for $M_k(A)$ a new empty modified concept,
- insert the actual modality set,
- connect $M_j(A)$ to all instances and modified concepts referred to by the premise and those which are already referred to by $M_j(A)$,
- activate the functions like in RULE1

Fig. 8. Rule for the data-driven modification of concepts.

RULE5: *Model-driven modification of concepts*

IF for a concept A or a modified concept $M_j(A)$ an instance I(B) or a modified concept M(B) exists and for a modality set of B there exists a link $B \overset{con}{\rightarrow} A$ or $B \overset{part}{\rightarrow} A$

THEN create new modified concepts $M_k(A)$ out of A or $M_j(A)$, respectively, as follows:

- create for $M_k(A)$ a new empty modified concept,
- connect $M_k(A)$ to all instances and modified concepts referred to by $M_j(A)$,
- activate for $M_k(A)$ the attached functions of A and B in the following sequence:
  - o inverse computation of attributes of B, which have an attribute of A as an argument,
  - o inverse judgment of relations of B, which have an attribute of A as an argument,
  - o inverse judgment of links of B, which have A as the goal node,
  - o functions of A like in RULE1

Fig. 9. Rule for the model-driven modification of concepts.

values for the tyre of a jeep which are tested for this link in the related judgment function. Analogous to that, if a judgment of relation tests the requirement 'Bodywork above Tyre', the knowledge of the location of *I*(Bodywork) is used to restrict the possible location of Tyre. This is done by the inverse judgment of relation. The inverse attribute computation works in a similar way (see Section 2.1).

The estimation of possible goal concepts is very important for an efficient analysis. Thereby, only a part of the knowledge base has to be investigated and the expectations of the model can be immediately incorporated in the analysis. The establishing of these goal concepts is possible due to initial segmentation results or due to expectations on an actual analysis. RULE6 in Fig. 10 describes this estimation of goal concepts.

The six rules above precisely define the creation of instances and modified concepts and allow a problem-independent utilization of knowledge.

```
RULE6: initial estimation of analysis goals
IF     for a concept A values are known for an attribute
       OR
       due to expectations inside an actual situation a concept A is given as a
       possible analysis goal
THEN   create modified concepts M_k(A) as follows:
       • create for M_k(A) a new empty modified concept,
       • insert the known values,
       • activate for M_k(A) the functions of A in the sequence of RULE1 with
         exception of the attributes with known values
```

Fig. 10. Rule for initial estimation of goal concepts.

The goal of an analysis process is not the creation of an arbitrary interpretation but the efficient computation of the optimal interpretation in the sense of 4. To reach this goal the following control algorithm determines which rule should be applied to which data in a given situation.

3. A problem-independent control algorithm

Besides the representation of task-specific knowledge the flexible and efficient use of the available knowledge sources is necessary for the automatic interpretation of sensor signals like images or continuous speech. Dependent on the flow of information and the activity through the representational layers the data-driven (bottom-up) and model-driven (top-down) strategy are the two basic control paradigms. Unfortunately, the right way strongly depends on a specific task-domain. For a problem with unambiguous results of preprocessing a data-driven analysis is suitable, because this leads to a small number of competing interpretations. On the other hand, a model-driven strategy is efficient for applications with small and/or unambiguous knowledge base, because many hypotheses, incompatible with the model, can be excluded in an early state of the analysis. Otherwise, a mixed top-down and bottom-up strategy should be preferred using both the restrictions and expectations of the knowledge base as well as the data from preprocessing. Therefore, a problem-independent control algorithm must have both data-driven and model-driven control structures which can easily be combined to any mixed strategy.

Figure 11 shows an outline of the general control algorithm which offers both data-driven as well as model-driven control features. By an easy combination of these structures a broad variety of control strategies can be designed yielding a good adaptation to a specific task-domain. The control strategy is explained by the segmented image of Fig. 12. It contains hypotheses for circles, polygons, trapezoids and rounded rectangles. Besides the correct hypotheses, one incorrect polygon, circle and trapezoid were detected.

As the image and speech signals are ambiguous, competing instances and thus competing interpretations are calculated. To focus on the most promising interpretation the A^*-algorithm is used to direct the analysis. Every node in the search space represents one consistent (partial) interpretation of the sensor signal. That means, every node n represents the complete knowledge base adapted to the signal by the instances and modified concepts related to n. Therefore, the search space is initialized by the root node n_0, and the set OPEN, which contains the active nodes, is the empty set. Then, as starting points of the analysis initial goal concepts have to be selected. Dependent on the level of abstraction a more data-driven or a more model-based strategy is first performed. For example, by

```
initialize the search space by the root node $n_0$;   OPEN := $\emptyset$
select initial goal concepts $C_i$ due to expectations or initial segmentation results
FOR all $C_i$
    apply RULE6 to $C_i$
    FOR every created modified concept $M_k(C_i)$
        generate one successor node $n_{ik}$ of the root node $n_0$ and insert $M_k(C_i)$ in $n_{ik}$
        judge $n_{ik}$ due to an optimistic estimation for the costs of an interpretation of the
        signal and bring $n_{ik}$ to OPEN
WHILE OPEN $\neq \emptyset$
    remove the best judged node $n$ from OPEN
    IF     (1) the analysis goal is reached for node $n$
    THEN   stop analysis as $n$ contains the optimal interpretation of the sensor signal
    ELSIF  (2) a new instance can be created
    THEN   create a new instance by RULE1-3 and generate data driven modifications up to
           the goal concept by RULE4
    ELSIF  (3) all objects of $n$ are instances but the level of interpretation is not abstract
           enough
    THEN   estimate bottom-up new goals due to the paths in the knowledge base
    ELSIF  (4) a new model driven modification is possible
    THEN   create top-down a new modified concept by RULE5
    ELSIF  (5) there exist not interpreted signal areas, although $n$ contains a complete
           interpretation of an appropriate level
    THEN   due to the model, incorporate specializations by RULE1-4 or create a modified
           concept for an optional link by RULE5
```

Fig. 11. An outline of a problem-independent control algorithm.

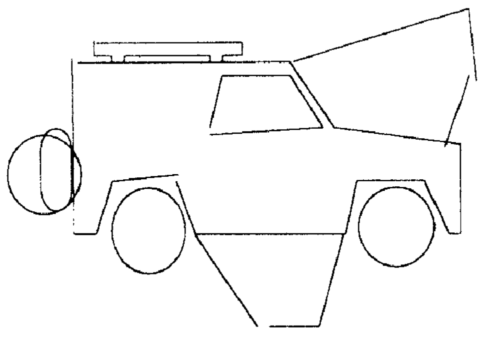


Fig. 12. Segmented image of a jeep.

the concept Polygon (most concrete level) the sensor data can be immediately incorporated in the analysis. This is done by the instantiation of Polygon. Such an instance I(Polygon) represents a concrete polygon found in the sensor data. On the other hand, an initial goal concept Car causes a model-driven strategy, as the expectations of the model determine the further processing. RULE6 is applied to every goal concept causing modified concepts $M_k(C_i)$. Every such modified concept is regarded as a competing hypothesis and therefore it is inserted in one successor node of n_0. To process only promising interpretations every search tree node n is judged on the basis of the modified concepts and instances collected in n. The initial nodes n_{ik} only contain one modified concept so that the judgment of n_{ik} bases on the judgment of that modified concept and on an optimistic estimation for the costs of a complete interpretation. As the judgment problem is for the most part problem-dependent, a treatment of that problem would exceed the scope of this article. An example for a judgment function in the task-domain 'speech understanding' is given in [47] and for 'object recognition' in [33]. In our small example we select

Polygon as the single initial goal concept and create M_1(Polygon), so that only one node n_1 is in OPEN.

After that initialization phase, the A^*-algorithm begins to work. While OPEN is not the empty set, the best judged node n is selected for further processing. If the analysis goal is reached, i.e. the entire sensor signal is interpreted on a level abstract enough, then the analysis is finished and n contains the optimal interpretation. This is not fulfilled for node n_1, so that the next condition is tested. To guarantee a fast verification of expectations of the model due to segmentation results, the instantiation is preferredly activated. As Polygon has no parts and no concretes, instances to that concept can be created by applying RULE1 and RULE2. Dependent on the results of preprocessing, for every detected polygon in the image one instance and one successor node is created (see Fig. 13). They represent competing partial interpretations of the image. Generally, the instantiation is a data-driven process which creates a new interpretation of a signal area. In the following, every new search tree node is judged and inserted in OPEN.

For the next iteration we assume that n_2 is the best judged node. For that node the third condition is fulfilled and new goals are estimated bottom-up. By this data-driven process the high-level knowledge is incorporated in the analysis whereby the search process can be reduced. The new goals are based on expectations resulting from the knowledge base and on a verified partial interpretation. For that, starting from the old goal a connection via instances and modified concepts is built up to the new goal following the inverse links part-of and concrete-of in the network. Different paths in the network result in competing search tree nodes. Figure 14 shows two competing search tree nodes after the estimation of new goals. In node n_5 I_1(Polygon) is interpreted as a body-work of a car, while for n_6 I_1(Polygon) is interpreted as a roof rack. These hypotheses are expressed by the two modified concepts M_1(Car) and M_2(Car) and by the two instances I_1(Bodywork) and I_1(Roof_rack). After the first estimation, the new goals usually do not belong to the most abstract level. They only represent intermediate goals which are verified in a model-driven manner. After the verification of these intermediate goals, new goals in higher levels are estimated. This alternating process is repeated until the desired level of abstraction is achieved. Dependent on the length of the estimated path more or less knowledge of the model is used for further processing. Correspondingly, a more or less model-driven strategy is designed.

In the next iteration, for the (best judged) node n_5 a model-driven modification is possible. Due to the new modified concept M_1(Car) RULE5 is applicable to Window, Roof_rack and Tyre. To guarantee an efficient analysis only that concept is modified which is referred to by an obligatory link and which can most easily be instantiated. Therefore, every concept has a priority score depending on the length of a path from that concept to an initializing concept. For an exact definition of the

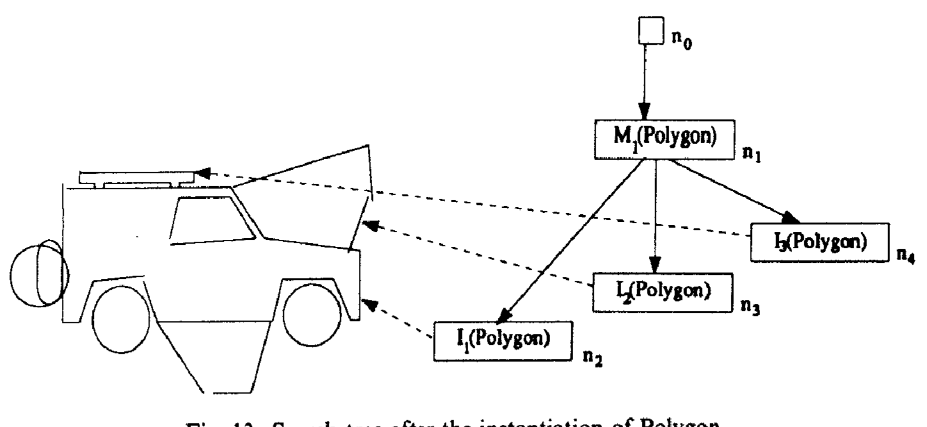


Fig. 13. Search tree after the instantiation of Polygon.

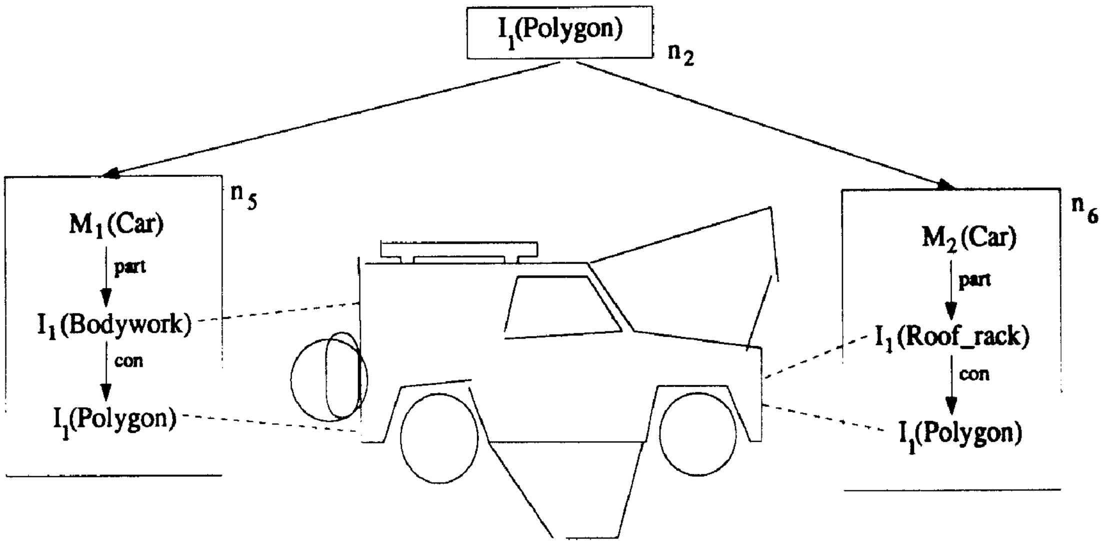


Fig. 14. Content of search tree nodes after the estimation of new goals.

priority of a concept see [46]. Besides this problem-independent measure, for every link a precedence can be defined expressing task-specific knowledge. For example, if the goal concept A of a link can be instantiated with high certainty, then a good judged instance $I(A)$ confirms the so far generated partial interpretation. Contrarily, by a worse judged instance the interpretation can be rejected. By this discriminatory role, a model-driven modification of A has a high precedence. If the link Car $\xrightarrow{\text{front\_tyre}}$ Tyre has highest procedence a new modified concept for Tyre is generated. By that process all restrictions of M_1(Car) are propagated into M_1(Tyre). This is done by activating the inverse functions of Car for M_1(Tyre) due to RULE5. Figure 15(a) shows the new generated search tree node and the restriction of M_1(Tyre) resulting from the position of I_1(Polygon) in the image. In the next step, these restrictions can be propagated into M_1(Circle) (see Fig. 15(b)).

If node n_8 is selected for further processing, M_1(Circle) can be instantiated. Due to the restriction propagated in M_1(Circle) only the circle in the restricted area is admissible. Therefore, only one instance and one search tree node is created (see Fig. 16). However, in a pure bottom-up process, due to the three hypotheses for circle, three competing instances and three competing search tree nodes had to be generated. Besides the increased efficiency, errors of the segmentation process can be corrected by that model-driven instantiation of a concept. For example, if a partial interpretation predicts a segmentation primitive in a certain area of the sensor signal and the initial segmentation process did not find such an element, then a special procedure for that primitive can be activated. By that strategy, the more costly procedures are only activated in small areas of the sensor signal by request of high-level information.

To allow a propagation of the constraints arising from I_1(Circle) RULE1–4 are applied to those concepts connecting M_1(Circle) with the goal of node n_8 (M_1(Car)). By that process, a new instance I_1(Tyre) and a new modified concept M_2(Car) are created (see Fig. 16). M_2(Car) contains all restrictions due to I_1(Bodywork) and I_1(Tyre). These restrictions can be propagated in the next iteration of the control algorithm.

After the model-driven modification of Tyre (role rear_tyre), Circle, Window and Trapezoid and after the instantiation of the resulting modified concepts the concept Car can be instantiated (see Fig. 17).

If the analysis goal is reached for node n_{15}, the control algorithm stops. Otherwise, the last condition is fulfilled and optional links and

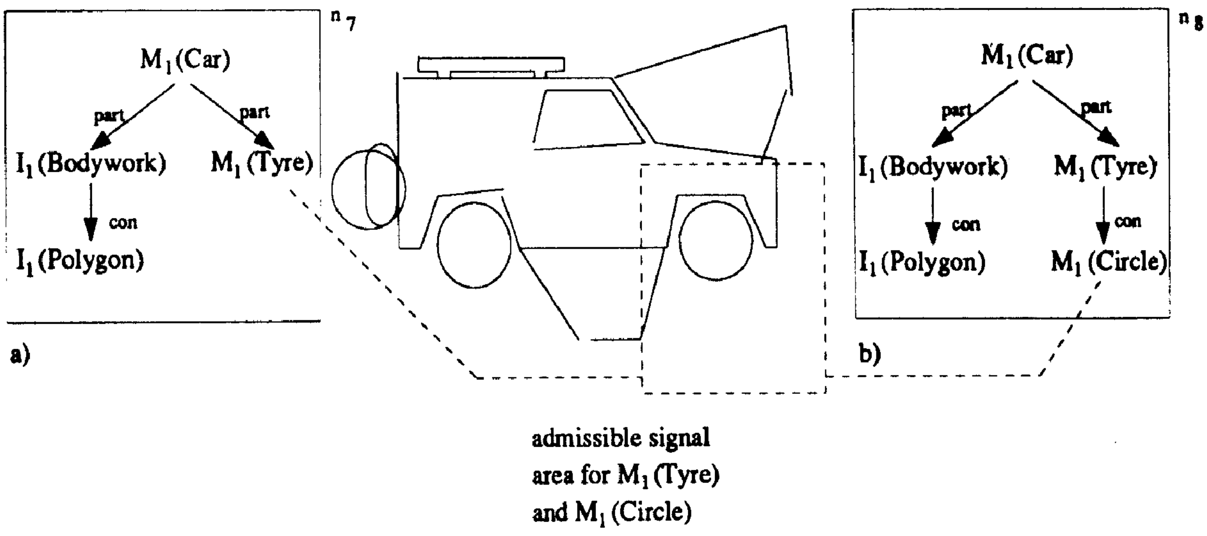


Fig. 15. Content of search tree nodes during the analysis.

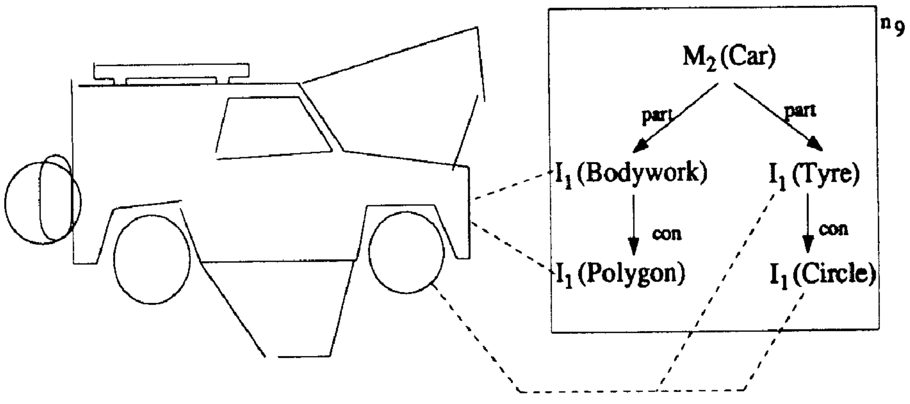


Fig. 16. Content of a search tree node during the analysis.

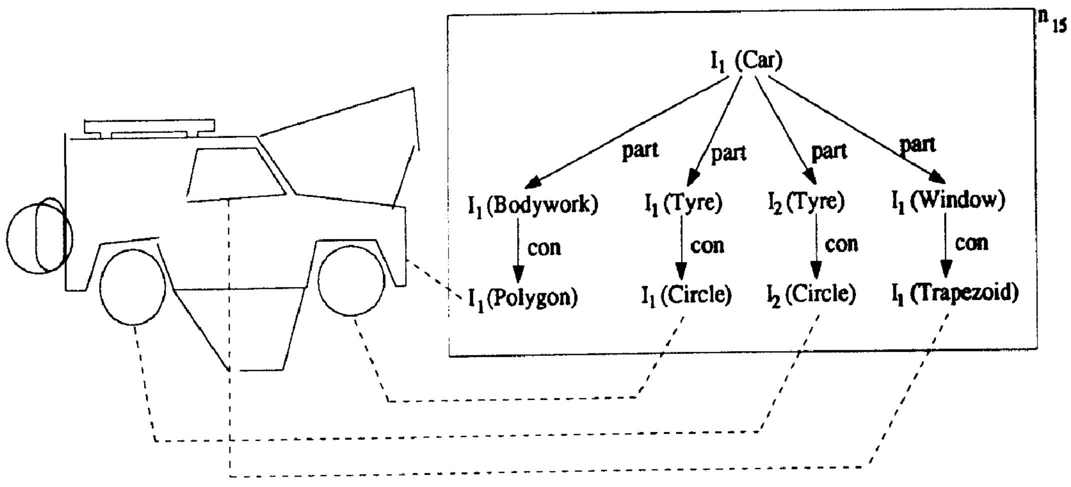


Fig. 17. Content of a search tree node during the analysis.

specializations are incorporated in the analysis. Due to the network of Fig. 2, one specialization and two optional links exist. As Cowcatcher only exists for Jeep and node n_{15} only contains an instance to Car, the related link is not regarded in that state of the analysis. The incorporation of the remaining concepts Roof_rack and Jeep may yield in competing interpretations, so that two competing search tree nodes are generated. One interprets the signal as a car with a not yet detected roof rack (see Fig. 18(b)) and the other one says there is possibly a jeep with known bodywork, window and tyres. As Spare_tyre is a context-dependent part and Cowcatcher is optional, a partial instance to Jeep was created (see Fig. 18(a)).

Assuming that node n_{16} is the best judged node in OPEN, the related interpretation is verified. After a top down modification of Spare_tyre and Rounded_rectangle, the corresponding modified concepts and Ip_1(Jeep) can be instantiated (see Fig. 19). If for n_{20} the analysis goal is reached, the control algorithm stops. Otherwise the optional link Jeep $\xrightarrow{\text{part}}$ Cowcatcher is regarded.

In the example only the basic properties of the control algorithm should be demonstrated. These are the data-driven interpretation in terms of the knowledge base (instantiation, goal estimation) and the model-driven generation of predictions out of the knowledge base (top-down modification, specialization, optional links). Dependent on the selection of initial goals and on the intermediate estimation of new goals, almost any strategy can be achieved.

The successful use in different task-domains (see Section 5) indicates the quality of the presented control algorithm. The obtained results show that the problem-independent control algorithm is able to handle totally different applications in an efficient manner.

More information about the control algorithm can be found in [18].

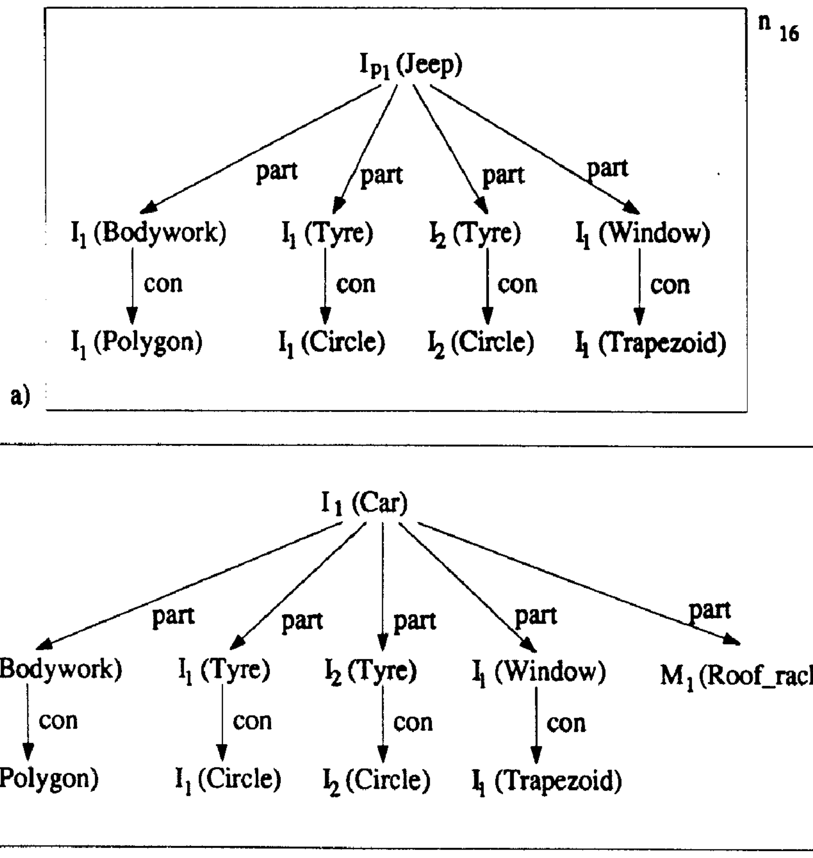


Fig. 18. Content of search tree nodes after the incorporation of specializations and optional links.

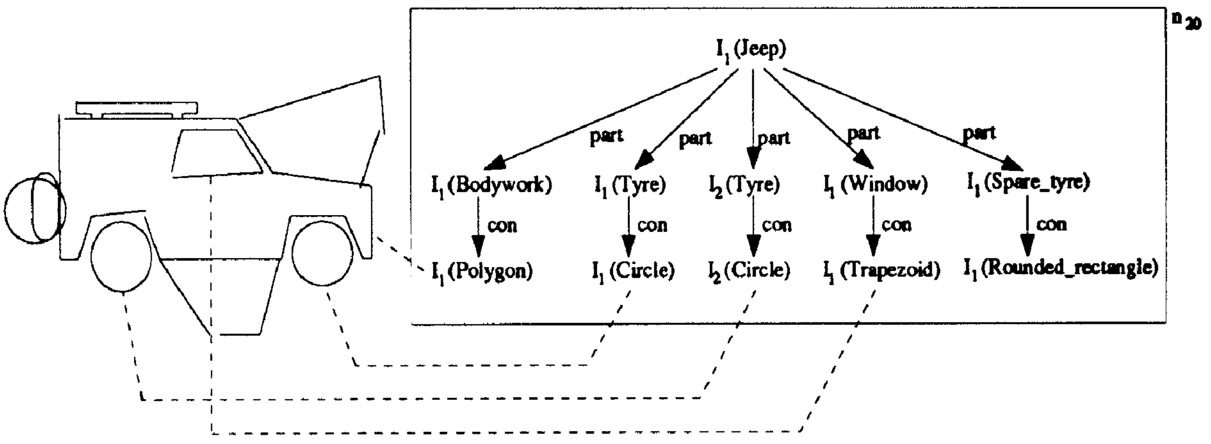


Fig. 19. Content of a search tree node with a complete interpretation.

4. The explanation facility ELK

One advantage of knowledge-based systems is that powerful explanation facilities can be provided. The use of such an explanation tool is twofold:

- It helps the system designer to develop, inspect and debug the knowledge base and the intermediate and final results of an analysis.
- The acceptance of the system is improved because the user can get explanations of what the system does, how it works, why its actions are appropriate, and when the limits of the system are reached.

In ERNEST complex (large and highly structured) knowledge bases must be explained. Therefore, we called the explanation facility of ERNEST ELK – Explanations for compLex Knowledge bases. Three parts of ERNEST have to be explained by ELK: the declarative knowledge, the procedural knowledge and the results.

To be able to give flexible explanations, the expert system needs a highly structured knowledge base [53], for example control knowledge ought to be stored explicitly [6]. The expert system also needs a deep model about the domain to provide justifications [4, 53]. ERNEST is highly structured. Therefore, specific explanations can be generated. However, some interesting knowledge cannot be represented in ERNEST in a declarative manner (for example, informations about the purpose of the whole system or of special parts). In these cases the system developer can integrate this knowledge in text files which will be shown by the explanation facility.

Many explanation facilities use a natural language approach. The user asks questions, typical are 'WHY' and 'HOW' questions which were first used in MYCIN [3]. In ELK we use a multimedia approach. Graphics are more efficient than natural language explanations with respect to showing the structure of the network and facilitating the comparison between different results. An example for such a graphic is given in Fig. 29. Natural language is used for our on-line help system. For the domain of image understanding the explanation facility can also provide the analyzed images and annotate them with the instantiated objects. For speech understanding speech output can be integrated.

We want to explain complex knowledge bases. For example, networks of the applications mentioned in Section 5 have about 100–250 concepts and 400–800 links. According to the principle 'divide and conquer' we divide an explanation in single explanation units. The access to further detail or support explanations is provided by pickable objects. Therefore, we get a hypermedia [21, 56] based system architecture.

Figure 20 shows the structure of ELK. In this section, the three modules of ELK – the extraction of data, the layout and the user interface – will be discussed. Because of the use of the programming

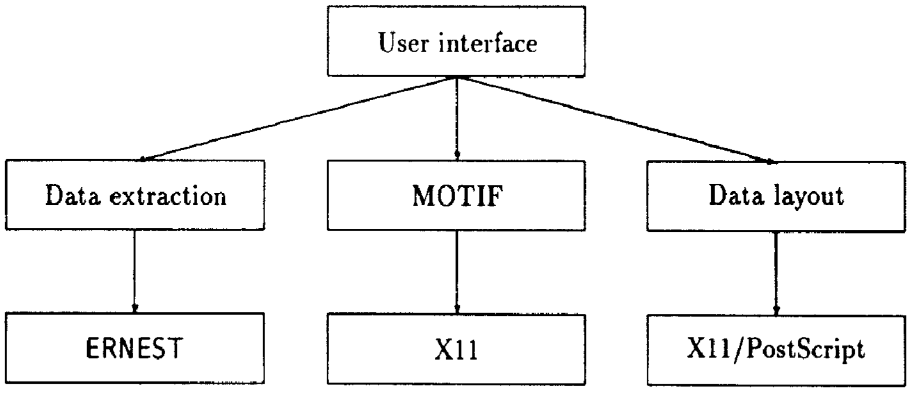


Fig. 20. The structure of ELK.

language C and the software packages MOTIF[1], X11[2] and PostScript[3], ELK is portable on many UNIX systems.

4.1. *The extraction of data*

The extraction module is the interface between ELK and ERNEST. Because complex knowledge bases must be handled, we cannot display all of the information in one single explanation unit of the size of a workstation screen. We must solve the problem of extracting the information in such a way that one single explanation unit is understandable.

We use seven principles to separate and compress the available information:

1. The development of explanation units and explanation sequences.
2. The separation of static knowledge and dynamically generated results.
3. The separation of different levels of detail.
4. The separation of different conceptional systems and topics.
5. The extraction of 'important' information.
6. The adaptation based on the user group and the user knowledge.
7. The adaptation according to the current interests of the user.

We already mentioned that we separate different explanation units. For some problems one explanation unit will be sufficient, but there also exist complex problems where you need a sequence or a combination of explanation units. For these complex problems we define explanation sequences to have a 'read thread' in the explanation.

Different explanation units and sequences must be provided for the explanation of the static knowledge in the system and the dynamic use of the knowledge. The first sequence gives an introduction to the system, the second one helps to check the results.

In an explanation one will first give an overview and then discuss details. Therefore, a separation of different levels of detail is needed. In ELK the user can get an overview of the network, then the graphic of one conceptional system, then details about one concept, i.e. about the attributes (see Fig. 21), and finally details about specific procedures. The overview of a network will be overloaded if the whole network of an applications with about 250 concepts and 800 links is supposed to be displayed at once. Therefore, we defined concept families summarizing similar concepts. The explanation of the results are structured in a similar way.

Within a level of detail different conceptional systems and topics can be distinguished. As mentioned before, *spec*ialization and *part* links are only

[1] MOTIF is a registered trademark of the Open Software Foundation.

[2] X11 is a registered trademark of the Massachusetts Institute of Technology.

[3] PostScript is a registered trademark of Adobe Systems Incorporated.

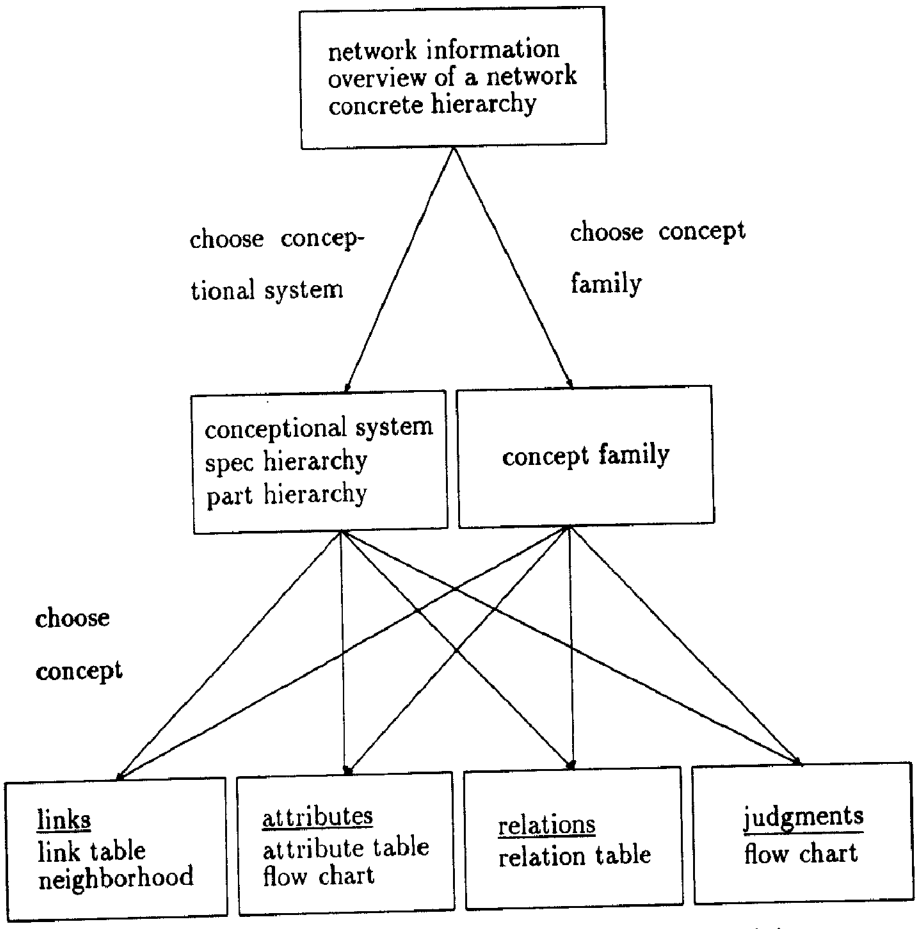


Fig. 21. Some explanation units of ELK for the declarative knowledge.

allowed within a conceptional system. For the connection between different conceptional systems exists the special link type *concrete*. In such a way the conceptional systems are clearly separated. Also each concept has different slots defining explanation topics like attributes or relations (see Fig. 21).

For each topic you can focus on the 'important' information. However, we need a definition of the 'importance' of the information. One sample of such a defintion in ELK is concerned with the search tree of ERNEST which is generated during the analysis of a signal. There exist search trees with several thousands of nodes, which makes a compression of this information very desirable. We provide a fisheye-view [11], the pruning of the tree, and the deletion of special node types. Figure 22 shows one example. Here paths with a low judgment and nodes with exactly one successor have been eliminated.

The explanation system is used by different users with different knowledge. A novice will need more information about the underlying ERNEST ideas and less information about confusing details of the knowledge base. On the other hand an experienced system developer needs quick and efficient access to every detail in the network without being distracted by information he already knows. For the intermediate steps between novice and expert it is necessary to assess the knowledge which the user has gained up to this point. To adapt the explanation sequence automatically to the user, a user model [15] is needed.

Finally, the explanation facility should provide an adaptation to the current interests of the user. Here the goals of the user play the decisive role. If he wants to build a new network, different explanations are needed depending on the knowledge acquisition module he uses (interactive or auto-

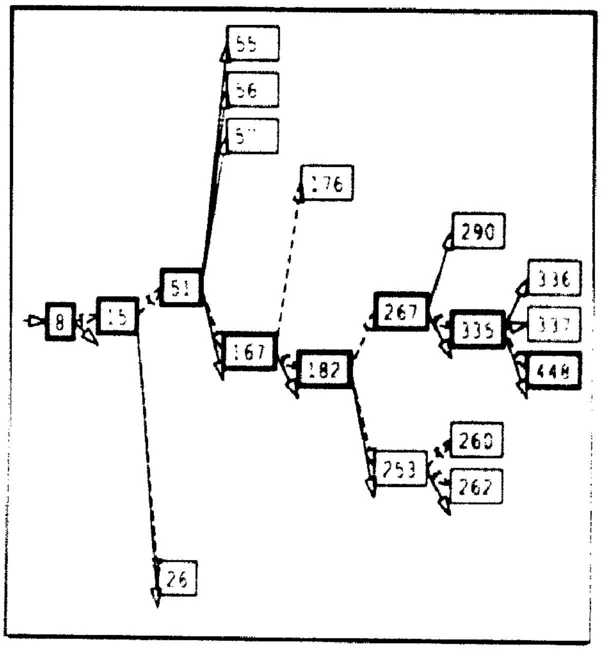


Fig. 22. A part of a search tree (dashed arrows indicate a sequence of nodes without branches, arrows with no goal node mark the deletion of paths with a low judgment).

matic). If he wants to get an overview of an existing network, he can also be interested in some details about attributes or judgments. Displaying only the desired information will lead to an easier acceptance of the system.

4.2. The layout

It was our goal to achieve explanation units which are easy to understand. Our approach follows the ideas and results presented by Bertin [1]. He describes which characteristics a graphic must have to provide answers to different questions in one single moment of inspection.

To achieve a well structured explanation unit one must first analyse the given information. The invariants of the information and the components which are changing must be found. The components must be examined to derive their number and to obtain the length and the structure of each component. There exist four structure types: associative (i.e. all red objects), selective (i.e. distinguishing between red and blue objects), ordered (i.e. small, medium, high) and quantitative (i.e. 1, 2, 3, . . .).

In ERNEST we have several criteria to structure the information, for example:

1. We defined for every link a *degree*, for example a concept with no part links gets the part degree 0 and concepts with parts get a part degree equal to the distance along part links to a concept with no parts. Therefore, the concepts can be ordered along their link degrees.
2. Many attributes depend on the results of other attributes. Therefore, the analysis procedures carry out an adequate execution sequence. These dependencies can be used for ordering the attributes.
3. The nodes of the search tree should be ordered according to their depth.

We have seven layout types based on how much and which structuring information is present:

- *Grids*. Here we have two discrete axes defining a grid for the plane. For example, the overview of a network has the two axes concrete and specialization degree (see Fig. 27 in Section 4.3).
- *Hierarchies*. Here we have only one discrete axis. The second dimension of the plane can be used to minimize the number of crossings or the length of the connection lines (see Fig. 28 in Section 4.3).
- *Trees*. Here we also have only one discrete axis, the depth of the tree, but it is easy to construct a graphic without crossings. To minimize the length of the connection lines every node is placed in the vicinity of its father (see Fig. 22).
- *Neighborhoods*. Here only the direct predecessors and successors are displayed, i.e. the length of the components is only 3. Therefore, it is possible to use the two dimensions of the plane and also the two diagonals and so we can represent four quantitative structures (see Fig. 23).
- *Diagrams*. Here the two dimensions of the plane are used to show a quantitative structure in the horizontal direction and an associative structure in the vertical direction (see Fig. 24).
- *Tables*. In a table both directions of the plane represent an associative structure.
- *Texts*. In a text also unstructured data can be provided.

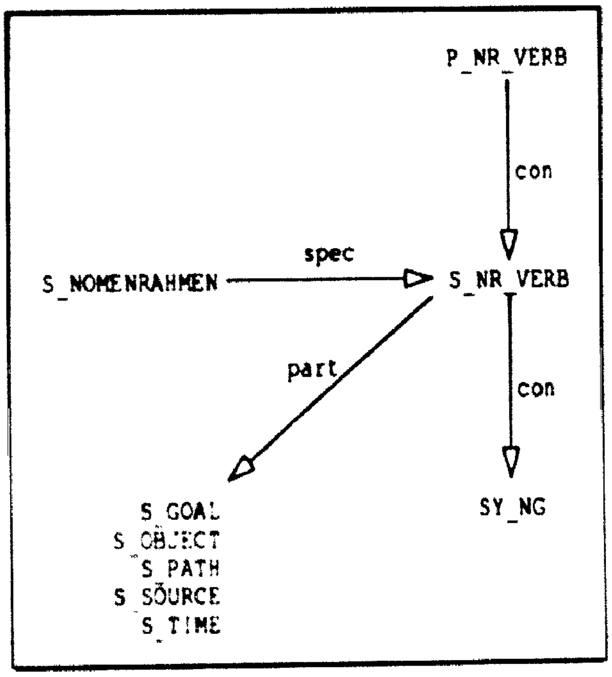


Fig. 23. A neighborhood in a knowledge base for a speech understanding problem.

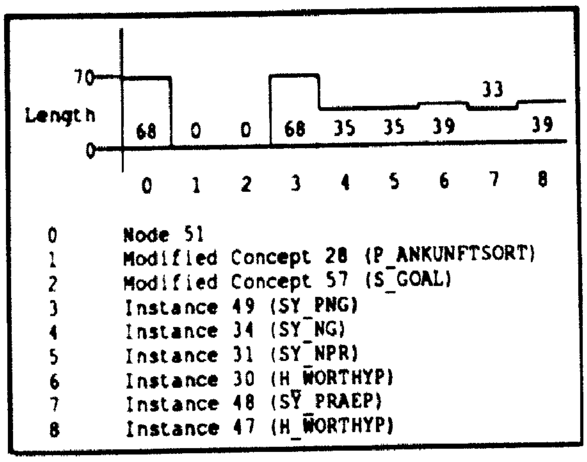


Fig. 24. A diagram of speech frames. The y-axis shows the number of speech frames interpreted by the instances and modified concepts in the node 51.

Besides these layout types, probem dependent explanation functions can be integrated. Here you can use the medium and the display method which is appropriate for your domain (also images, animations and speech output).

4.3. *The user interface*

The user can access the graphic with a command and a menu interface. For each command, for each menu, and for each ERNEST term a passive, context-sensitive help is provided. The objects in the graphics are pickable to allow a quick access to further details. All choices listed in Fig. 21 can be done by selecting the objects in the graphics.

Figure 25 shows the main menu of the interface. With each menu item a pulldown menu is associated. For each explanation unit there exists one menu item in one pulldown menu.

For example, you can select the net overview in the network menu. Now you get a parameter menu (Fig. 26) where you can select the options and parameters of this explanation unit. When you push the START button, the net overview is created (Fig. 27).

Now you can pick every ERNEST object in this graphic. You then get a popup menu where you can select the detail explanation you want. For example, you can select a concept of the highest concrete level and choose as explanation unit the part hierarchy of this level (Fig. 28). By selecting again an ERNEST object or by using the main menu you can access every explanation unit in an efficient way.

An experienced system developer has no problems to choose the explanation units which will be appropriate for his explanation goals. However, an inexperienced user needs some guidance. Therefore, in ELK special hypertexts are provided - called adaptive explanation strategies - giving a proposal for a sequence of explanations units. These hypertexts are created automatically using knowledge about explanations, the user and the application represented in an ERNEST network.

5. Realized applications

The environment for knowledge-based signal understanding tasks as presented in the previous sections was used successfully for a number of applications. The first realized system DISS concerns the diagnostic interpretation of scintigraphic image sequences of the heart [42, 48]. Another medical application is the system AUDIGON [36].

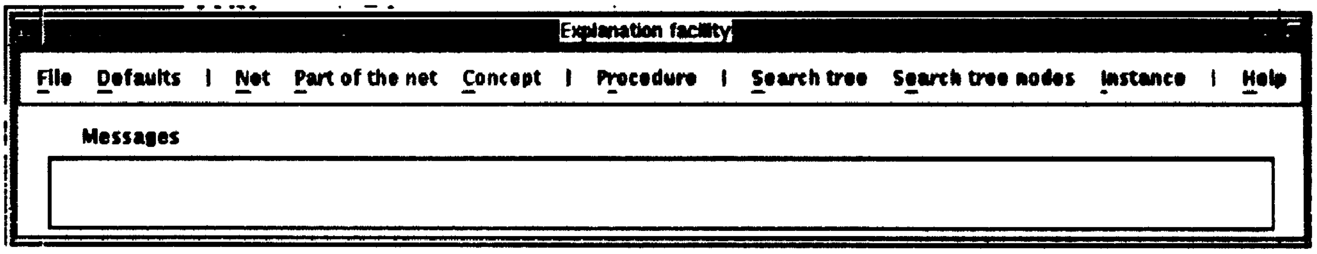
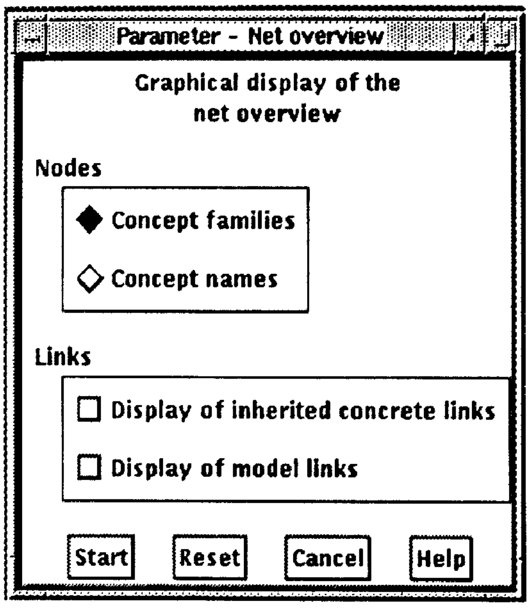

Explanation facility
File Defaults | Net Part of the net Concept | Procedure | Search tree Search tree nodes Instance | Help
Messages

Fig. 25. The main menu of ELK.



Graphical display of the net overview

Nodes

◆ Concept families

◇ Concept names

Links

☐ Display of inherited concrete links

☐ Display of model links

Fig. 26. The parameter menu of the explanation unit 'network overview'.

As well as in the system DISS, AUDIGON creates diagnostic interpretations on the basis of magnetic resonance images of the knee. The task is detection and validation of gonarthroses. Within a larger project for flexible production systems a component for recognizing industrial parts including the capability of performing and controlling a sequence of actions is under development. The task of this vision component is to control a cell in an assembly by analyzing images of its environment. It recognizes parts, evaluates their quality, inspects the environment, and controls the placement of the part [31, 33]. The other application is the system EVAR [35]. Here, the ERNEST environment integrates the various levels of a speech understanding and dialog system into one homogeneous architecture. In the following, the interpretation of image sequences of the heart and the speech understanding and dialog system are presented in more detail.

5.1. Scintigraphic image analysis

The concrete and the specialization hierarchy of the whole network is shown in Fig. 29. It integrates knowledge about objects, motility and medical evidence. Since there are about 150 concepts and 400 links for specializations, concretes and parts, Fig. 29 can only give a condensed view of the general structure of the network.

The lowest levels of the network are built up by the concepts INPUT SEQUENCE DESCRIPTION, respectively IMAGE SEQUENCE. INPUT SEQUENCE DESCRIPTION describes questions to the user about the scintigram sequence to be analyzed, that are the spatial and the time resolution and the name of the sequence. With this information an instance of IMAGE SEQUENCE can be created. Such instances cover both the original image sequence and the sequence after spatial median filtering. At the conceptional system which is characterized by OBJECT (level 2), concepts for the objects HEART, LEFT VENTRICLE, RIGHT VENTRICLE, and four anatomical segments of the left ventricle are defined. In the use projection LAO 45, these are the septal (SE), the inferio-apical (IA), the posterolateral (PL) and the basal (BA) segment. The part relationships between these concepts are obvious. Furthermore, all these concepts, except HEART, are described context-dependent from the concept they are part from. Among the main attributes are the contours and the volume curve of the objects. Level 3 describes form and size of the left ventricle and its

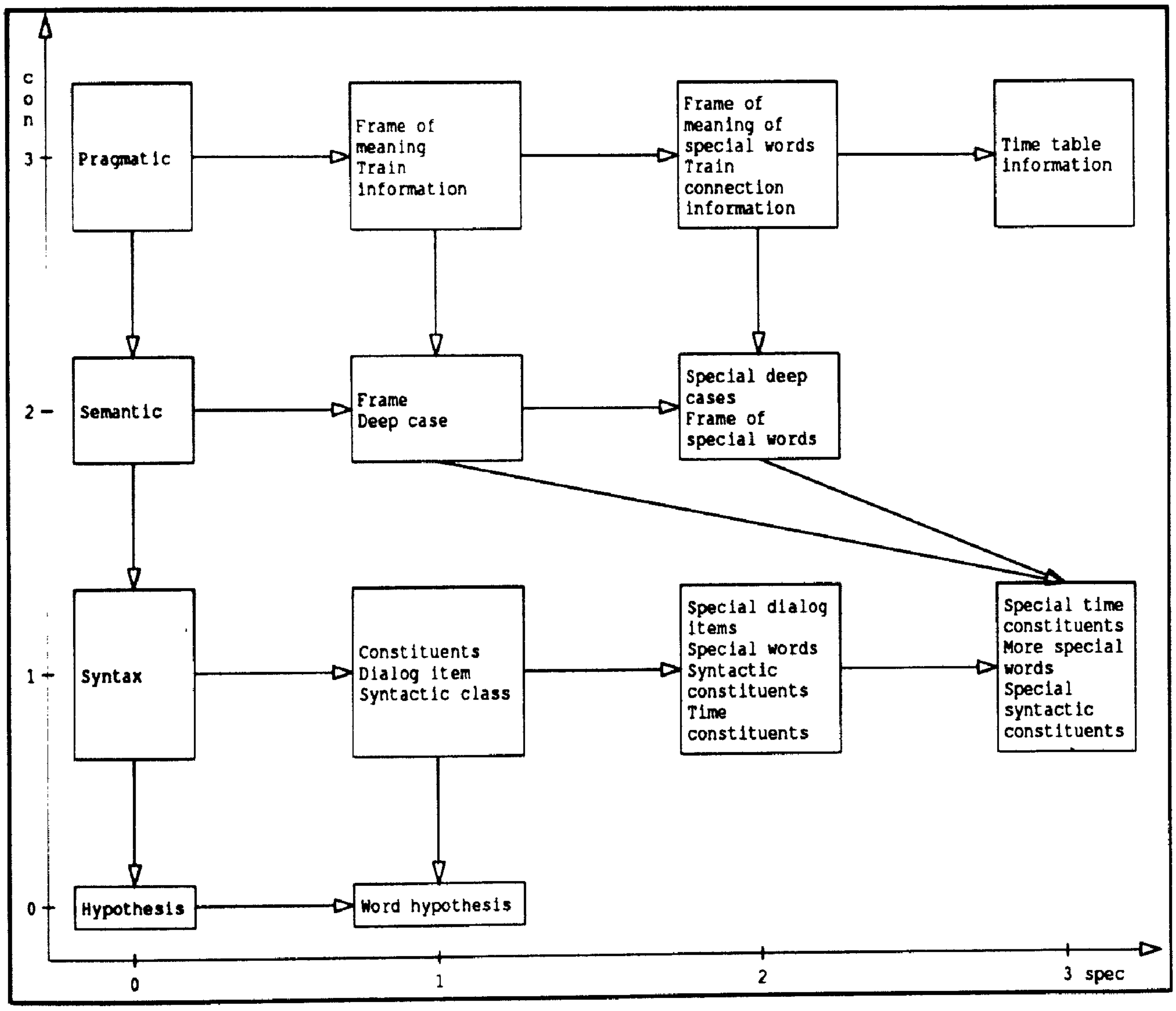

con
3
2
1
0
Pragmatic
Frame of meaning Train information
Frame of meaning of special words Train connection information
Time table information
Semantic
Frame Deep case
Special deep cases Frame of special words
Syntax
Constituents Dialog item Syntactic class
Special dialog items Special words Syntactic constituents Time constituents
Special time constituents More special words Special syntactic constituents
Hypothesis
Word hypothesis
0
1
2
3 spec

Fig. 27. An overview of a network which has been developed for a speech understanding system.

segments and basic motility. This may be used on level 7 to infer statements like 'left ventricle enlarged'. BASIS MOTILITY and MOTILITY are specialized for a cycle and its phases of contraction, expansion and stagnation. Those concepts are further specialized for motility descriptions of the left ventricle and its segments. One attribute of those concepts, which belong to the conceptional system characterized by BASIC MOTILITY, is the change of the area between two consecutive images. Furthermore, the direction and the strength of motility is obtained from the center of gravity of objects in consecutive images. Based on area changes the three phases of motility are characterized by fuzzy membership functions according to Fig. 30.

Using a set of anatomically possible volume curves, the best fitting of motion cycles is selected. The set of anatomically possible cycle types is specified and represented by the regular expressions

$$S^*C^+S^*E^+S^*E^* \text{ and } S^+, \qquad (5)$$

where C is Contraction, E is Expansion and S is Stagnation. Based on these expressions different cycle types are described. These types are generated out of the expressions above by splitting up the first expression into regular expressions which only

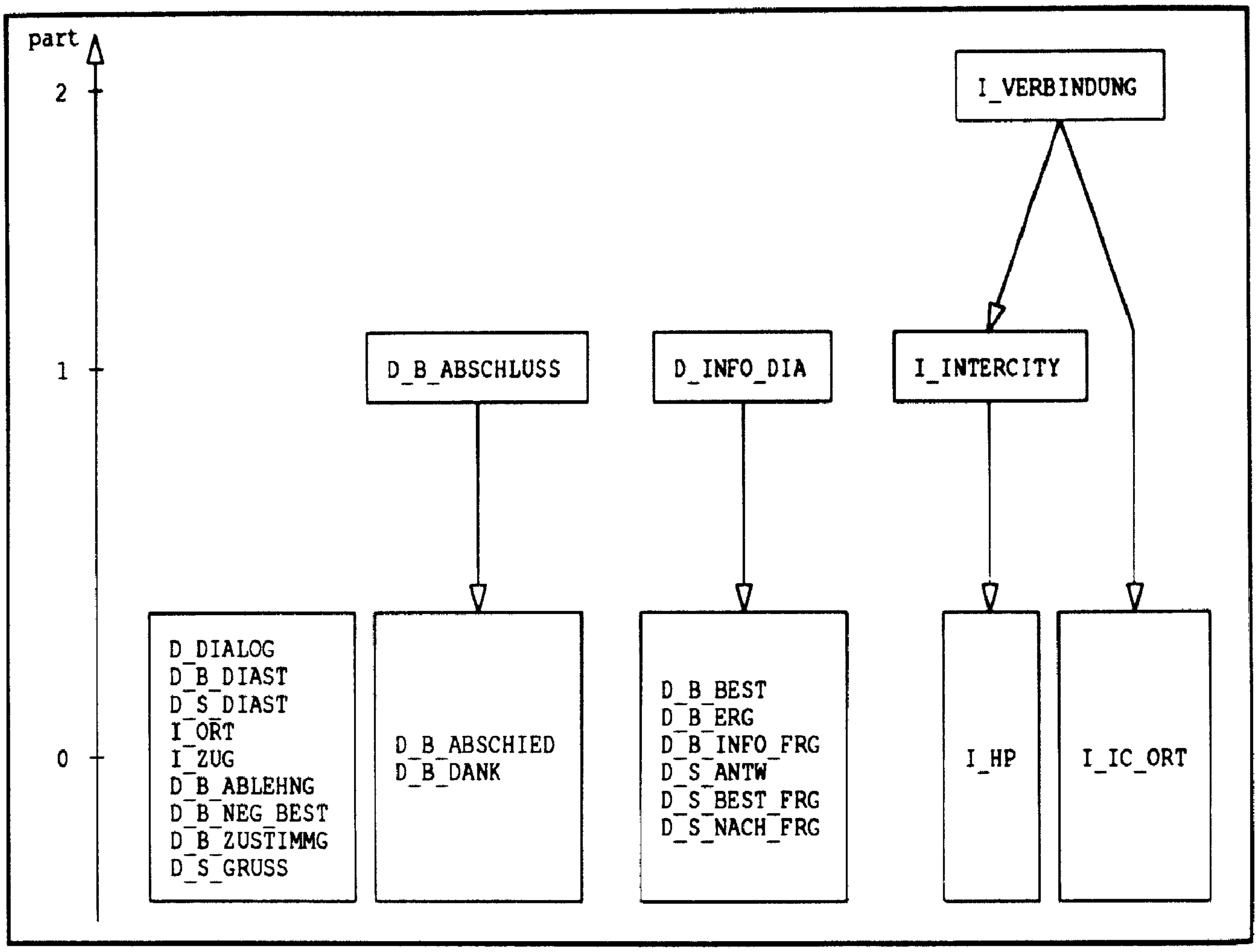


Fig. 28. A part hierarchy.

use the +-operator. The concepts at level 4 (MOTILITY) are defined by similar terms. The main difference is that level 3 (BASIC MOTILITY) always relates two consecutive images i and $i+1$, whereas level 4 describes larger entities, i.e., a contraction phase for images 1 to 7. Because of this description in larger entities, the concepts standing for contraction, stagnation and expansion in this conceptional system are defined to be context-dependent from the corresponding cycles. At this level no types are predefined. The cycle description is a verification of the description at the former level. Out of this verified cycle description the phases are extracted. Contrarily, all concepts which describe basic motilities are independent from a larger context. Several competing alternative descriptions of the motility of anatomical objects may arise during these steps. Level 5 gives a description of the motility of the left ventricle in medical terms. The general statement is that an anatomical interpretation of the motility of the left ventricle during a heart beat consists of systolic and a diastolic phase, and these phases are further sub-divided into pre-ejection period, ejection period, endsystolic stagnation, fast filling period, iso volume expansion and slow filling period. Attributes are start time, end time and ejection fractions. The result of these interpretations is a unique segmentation of the cycle into the described phases out of one interpretation at the former level. The last two conceptional systems relate motility phases, form and proportions to statements about medical evidence. On level 6 local diagnostic motility descriptions are derived, i.e., hypokinetic

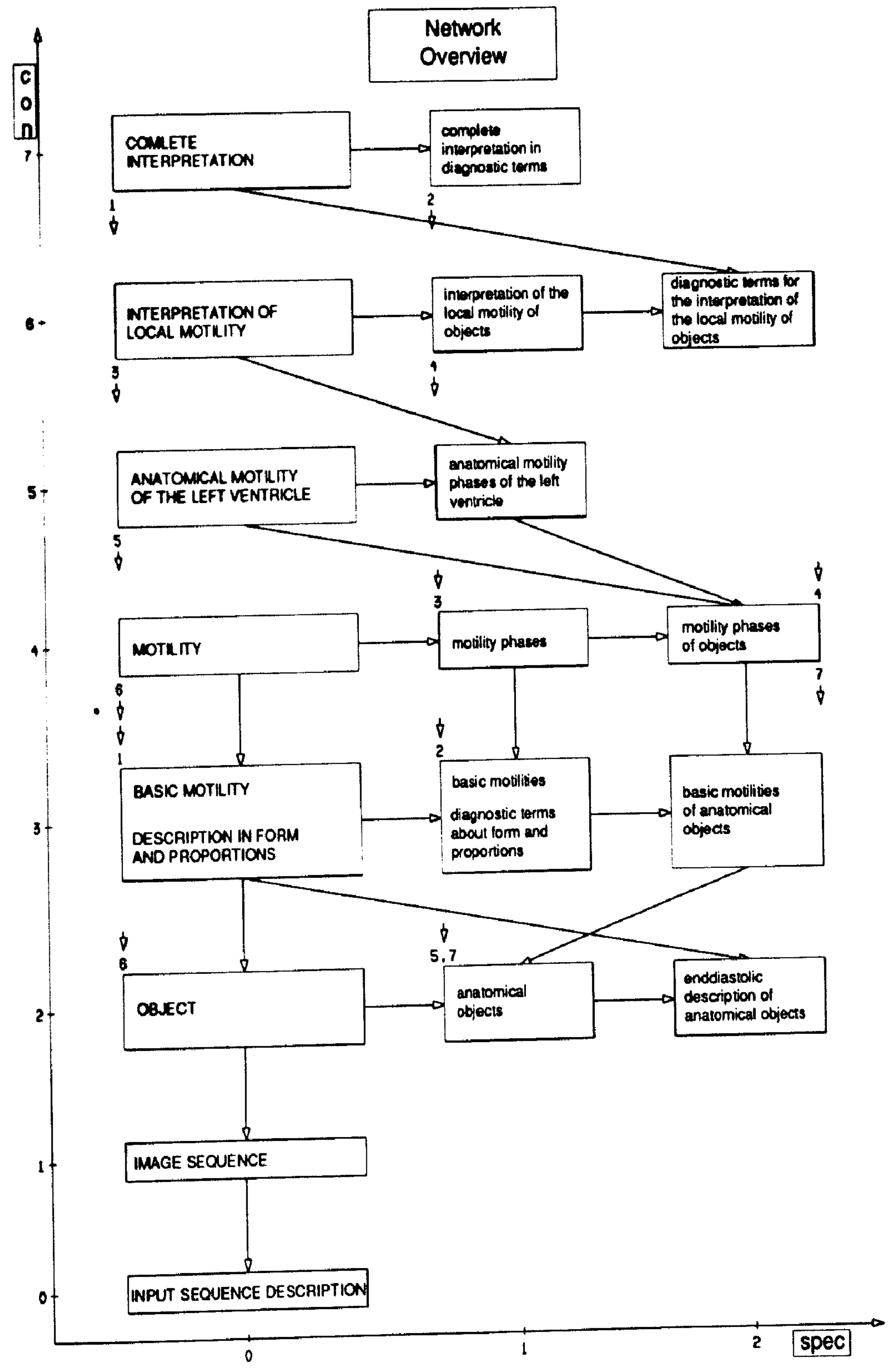

Network Overview
con
COMLETE INTERPRETATION
complete interpretation in diagnostic terms
INTERPRETATION OF LOCAL MOTILITY
interpretation of the local motility of objects
diagnostic terms for the interpretation of the local motility of objects
ANATOMICAL MOTILITY OF THE LEFT VENTRICLE
anatomical motility phases of the left ventricle
MOTILITY
motility phases
motility phases of objects
BASIC MOTILITY
DESCRIPTION IN FORM AND PROPORTIONS
basic motilities
diagnostic terms about form and proportions
basic motilities of anatomical objects
OBJECT
anatomical objects
enddiastolic description of anatomical objects
IMAGE SEQUENCE
INPUT SEQUENCE DESCRIPTION
spec

Fig. 29. Network overview of the system DISS.

postero-lateral segment or akinetic basal segment. For this purpose, the anatomically motivated segmentation of the left ventricle cycle is used together with the descriptions of level 4 to get the interpretations for the four segments and the left ventricle. On the top level the local diagnostic

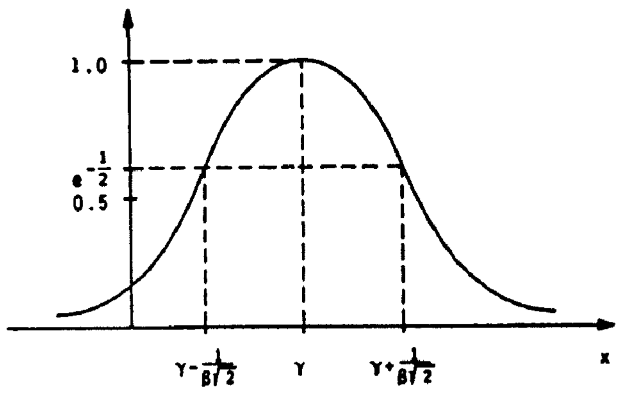


Fig. 30. Example of a fuzzy-membership-function.

interpretations are combined with each other and with the description of form and proportions. This allows us to model the complete diagnostic interpretations. Interpretations like ANEURYSMA are inferred but also local interpretations can be modified. For example, if a postero-lateral hypokinesis and an inferio-apical akinesis are confirmed with high certainty, it should be checked whether the first one is only a side effect of the second one, because both segments are mechanically coupled. If motility deficiency of the postero-lateral segment is strongest in the vicinity to the inferio-spical segment only the inferio-apical akinesis will be maintained.

A large amount of procedural knowledge is attached to the network which builds up the knowledge base of the system. In fact, the ratio of procedural to declarative knowledge is about 10 to 1 concerning the required storage space. Therefore, a full description is impossible here and probably not necessary because a good deal is fairly straightforward, for example, computation of areas, centers of gravity, lengths of contours, axes of an ellipse approximating the left ventricle, and so on. The most complex and most sophisticated algorithms are the detection of contours, the two-stage interpretations of the volume curves of the left ventricle and its four segments, and the inferences about medical evidence. In order to describe the analysis of an image sequence from the procedural points of view we use some snapshots on one example of an analysis process. The attribute values which will be shown are bound by those instances which are associated to the optimal path in the search tree. Low level methods are referred to in the three lowest levels of the network. With respect to the answers of the user, which are stored in an instance of the concept INPUT SEQUENCE DESCRIPTION, the input image sequence is transferred into an instance of the concept IMAGE SEQUENCE and immediately median filtered when creating this instance, Fig. 31.

Segmentation is done by creating instances of the concepts which belong to the conceptional system OBJECT. Detection of the heart and the left ventricle respectively is reported in [32]. The closed contour of the left ventricle is partitioned into the four anatomical segments septal, inferio-apical, postero-lateral and basal by using a priori knowledge about expected contour directions. Tests on image sequences gave the result that on the type of images provided by our medical partners a sufficiently reliable segmentation is achieved by the above procedure (about 90% correlation with the hand segmentation by a medical doctor). Additional details are available in [29, 43]. Each of these procedures is bound to the concepts for which they detect the boundary. Therefore the resulting contours are attribute values in corresponding instances. Figure 32 shows an image sequence with the contours of the left ventricle and four segments IA, PL, BA, SE.

Additional attributes in the concepts of this level are the volume curves of the objects. Figure 33 shows the volume curve of the left ventricle.

Interpretation of a heart cycle for each of the objects left ventricle (LV), IA, PL, BA and SE, is a two phase process executed on the levels 3 and 4 of the knowledge base. The basic idea is to use optimal search techniques [29, 37]. Image to image fuzzy membership values for stagnation, expansion and contraction are used as negative cost functions. Based on these costs, dynamic programming is used to find the optimal fit between the volume curve of each object and each cycle type of eq. (5). At the next level (MOTILITY) cycles are verified by introducing a refined alphabet, which subsumes

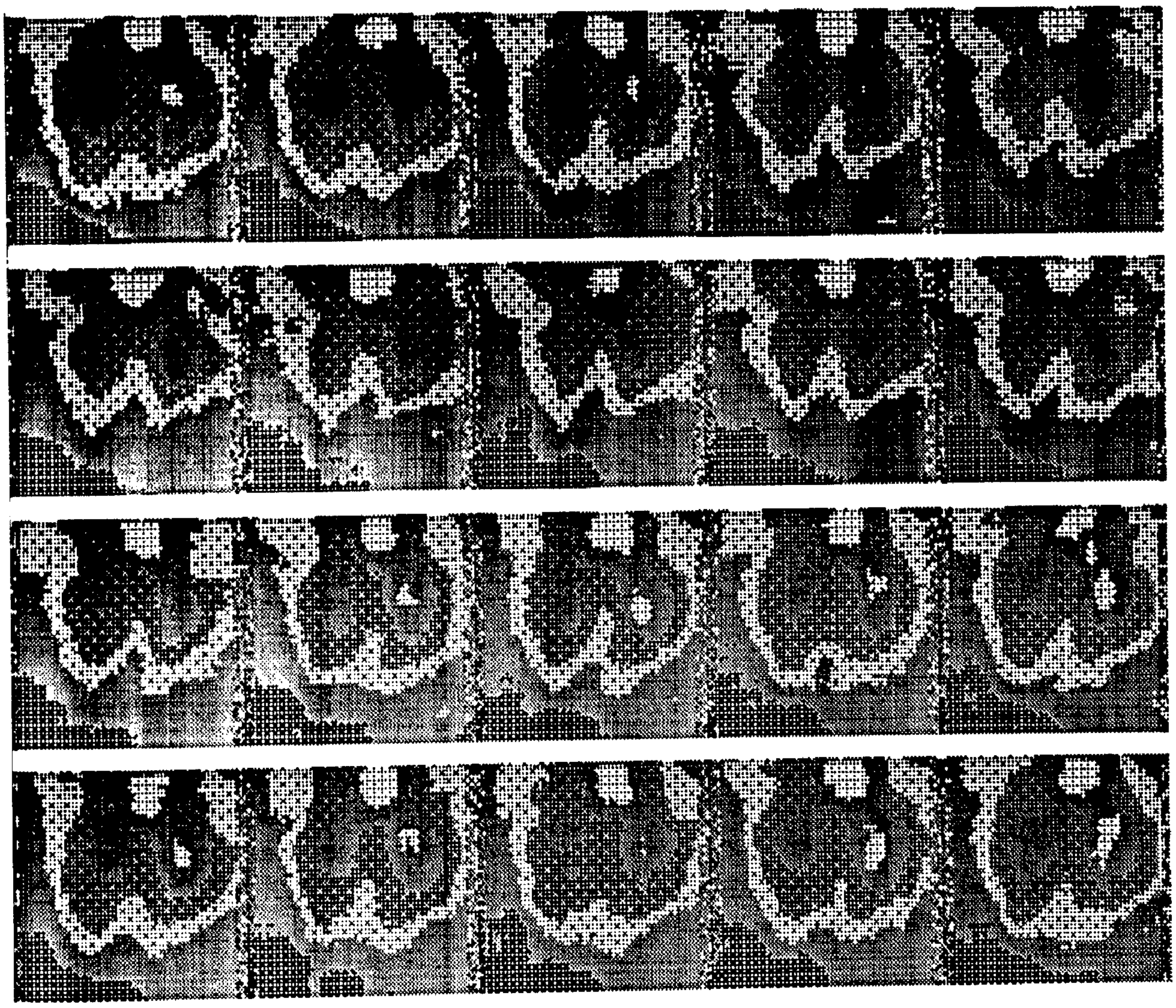


Fig. 31. Scintigraphic image sequence after 7 × 7 median filtering.

also symbols for the overlapping of different motility terms and for gaps inside one phase. These somewhat relaxed interpretations are generated in order to allow some flexiblity in later processing steps. Results of such a cycle description for the left ventricle is given by

$$C\,C\,C\,C\,[CS]\,[CS]\,S\,[SE]\,E\,E\,E\,E$$

$$[E-]\,E\,[SE]\,E\,[E-]\,E\,K\,K.$$

Overlapping symbols are denoted by the two elementary terms. This multilevel parsing procedure was presented in more detail in [44]. Out of these descriptions the motility phases are calculated with normalized start and end points (in the time range 0.0 to 1.0). Because most diagnostic terms can only be evaluated with respect to the global motility behaviour of the left ventricle, an intermediate level, called anatomical motility phases of the LV, is introduced. Here, the cycle of the LV is divided into the medical terms SYSTOLE, DIASTOLE, and in a second level into Pre-Ejection-Period, Ejection-Period, Fast-Filling-Period, Iso-Volume-Expansion and Slow-Filling-Period. This segmentation is based on the cycle description of the LV at the former level. For the calculation of each phase patterns in the former cycle descriptions are searched and scored with special fuzzy membership functions, which reflect a priori knowledge about the start time, the duration and the ratio between minimal and maximal area of the LV in the corresponding phase. Such ratios, which are also called

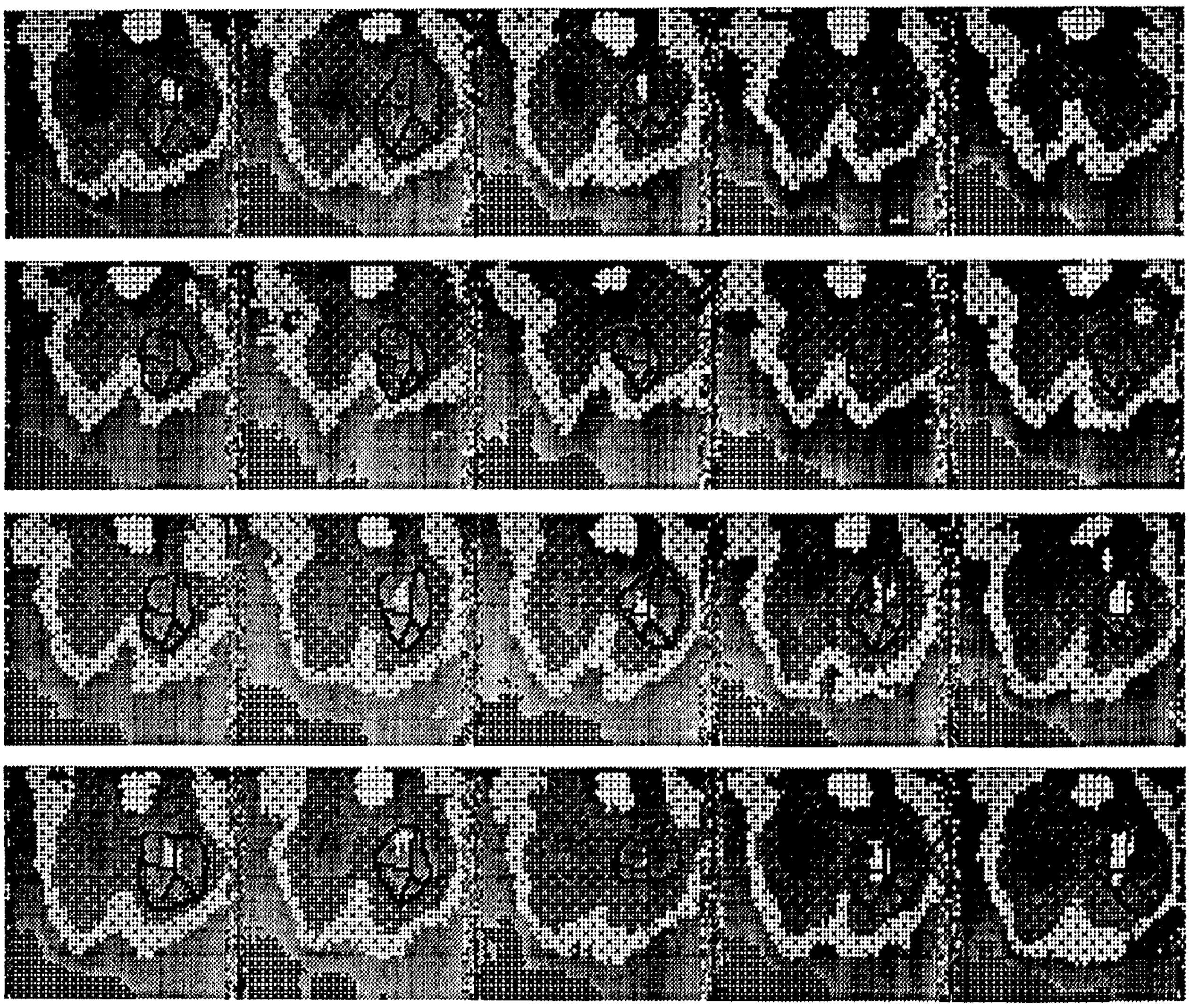


Fig. 32. Contours of the left ventricle and its four segments.

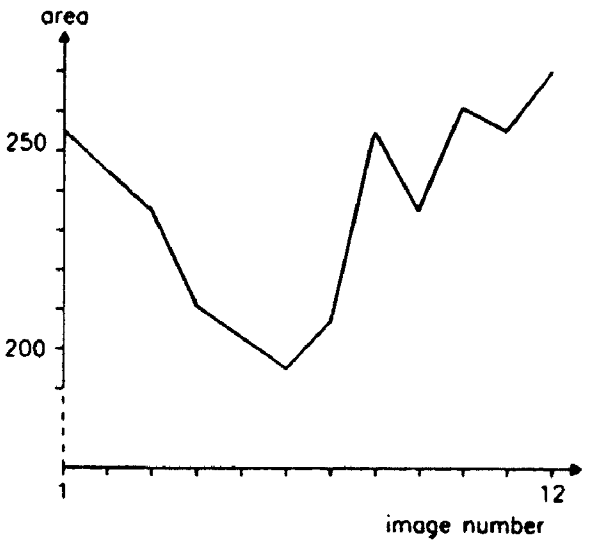


Fig. 33. Volume curve of the left ventricle (LV).

ejection fractions (ef), are themselves parameters for the diagnostic evaluation at both top levels in the network. Presently the system knows about 45 individual diagnostic interpretations which are represented by concepts at the levels 3, 6 and 7. These are of the type 'left ventricle is normally beating', 'the left ventricle is deformed' or 'the basal segment of the left ventricle is nearly motionless'. The certainty of each statement is measured by a certainty CF which is computed by fuzzy algebra. For example, given a rule of inference of the form

If (neg A) $\vee$ ($B \wedge C$) **then** D

the certainty factor of D is computed from those of A, B and C by

$$\mathrm{CF}(D) := \max(1-\mathrm{CF}(A), \min(\mathrm{CF}(B), \mathrm{CF}(C)))$$

or, in short hand notation,

$$\mathrm{CF}(D) := (\mathrm{neg}\,\mathrm{CF}(A) \vee (\mathrm{CF}(B) \wedge CF(C)).$$

For the very top level, i.e. for COMPLETE INTERPRETATIONS, A, B, C and D in the equations above stand for names of concepts in the knowledge base. Since there is an obvious one-to-one correspondence between the rule for deriving a conclusion and the CF of this conclusion, it is sufficient to state only the equation for the CF. The rules for complete interpretations at level 7 use concepts on level 6 (INTERPRETATION OF LOCAL MOTILITY) and/or level 3 (DESCRIPTIONS IN FORM AND PROPORTIONS) as arguments. One of the complete interpretations which is only based on local motility descriptions is akinetic motility (AKIN). The CF is given by

$$\begin{aligned}\mathrm{CF(AKIN)} := \mathrm{CF(LV\_WEAK)} & \\ & \wedge (\mathrm{CF(AKIN\_AI)} \\ & \vee \mathrm{CF(AKIN\_PL)} \\ & \vee \mathrm{CF(AKIN\_BA)} \\ & \vee \mathrm{CF(AKIN\_SE)}).\end{aligned}$$

That means an akinetic motility is stated if at least one of the four segments shows akinetic motility and the global left ventricle has a weak motility. Things are somewhat more complicated on level 6 because on this level descriptions have to be derived from motility phases. At this level the motility of individual segments of the left ventricle is diagnostically interpreted. This is done with the help of two functions t and u. t measures the overlapping duration between motility phases of the objects as achieved at level 4 and the anatomical motility phases of the left ventricle at level 5. u classifies parameters like the ejection fraction or the amount of stagnation during a cycle in accordance with objects and deseases. The result of applying some is again a fuzzy membership value. The functions u are derived from general medical knowledge [49]. With the functions u and t the CF's for individual diagnostic terms are determined.

The first operational version of the realized system has about 4 MB of code. Tests on about 20 image sequences having validated diagnostic evaluations by medical doctors showed that no wrong descriptions were given by the system; however, in about 10% of the cases the system could not give any description.

5.2. Speech understanding and dialog

The aim of the system EVAR is the automatic understanding of continuous German speech and the handling of an inquiry dialog in the task-domain intercity train connections. To have a structure where linguistic expectations could be used not only for the interpretation but also for the recognition process all kinds of knowledge are integrated in a homogeneous knowledge base. This allows an easy constraint propagation throughout the layers 'dialog', 'pragmatics', 'semantics' and 'syntax'.Figure 34 gives an overview of the knowledge base.

In the lowest level, the concept H_WORDHYP builds the main interface between word recognition and linguistic analysis. Top-down, the request resulting from the constraint propagation process are given to the acoustic module. Bottom-up, word hypotheses of the acoustic module are incorporated into the linguistic analysis. Besides this communication, the acoustic verification of word chains created during the linguistic analysis builds another interface between the recognition and understanding module. The syntactic level contains concepts for all syntactic classes (i.e. noun, verb, adjective), for syntactical constituents (i.e. noun phrase, verbal group), and for times (i.e. time of day, date). Each concept for a syntactic class has a concrete link to the concept H_WORDHYP. Larger syntactical units are built up by the syntactic classes and/or by simpler syntactical units (i.e. prepositional phrase $\xrightarrow{\text{part}}$ preposition, noun phrase). The semantics is based on the valency theory (see, i.e., [54]) and the case theory [9]. This level contains concepts for deep cases, verb frames and noun frames. The concepts for the deep cases

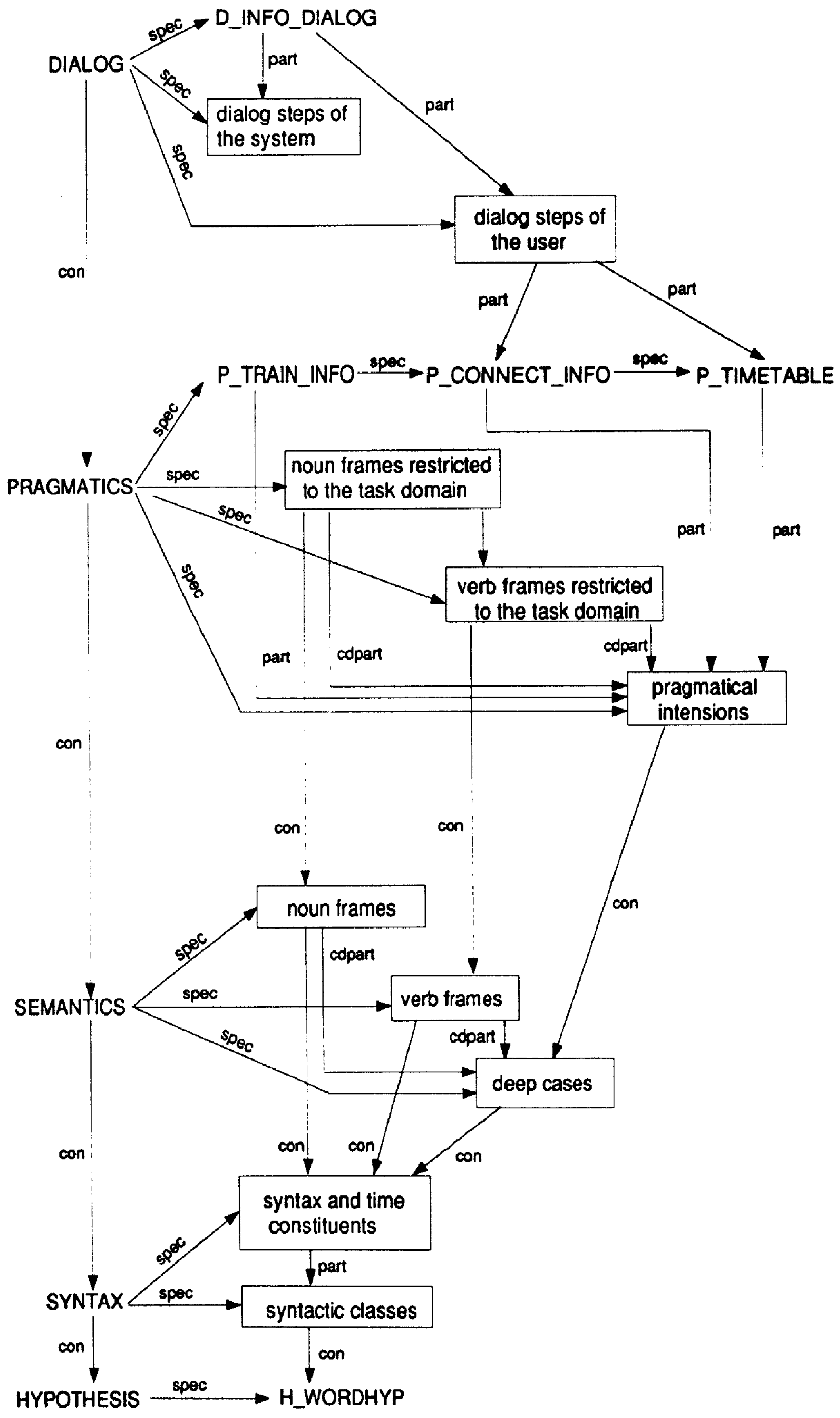


Fig. 34. Network overview.

are connected with the syntactic level by concrete links. The deep cases are context-dependent parts of the related frame. For each meaning of a verb or noun a modality set exists defining the obligatory and optional deep cases. The pragmatic level reflects the task-domain 'intercity-train-information'. Actually, concepts are modeled for the pragmatic goal concepts P_CONNECT_INFO and P_TIMETABLE, for the frames restricted to the task-domain, and the pragmatic intensions reflecting the interpretations of the deep cases in the application (i.e., P_DESTINATON, P_FROM_TIME). The highest level contains concepts defining the admissible dialog steps of the user and the system. Actually, the user steps request, addition, correction and confirmation are allowed.

The system can request detail, can give timetable information and can ask for a confirmation or for a repetition of the last utterance.

To use extensively the expectation of the linguistic knowledge base a partial interpretation is not extended by a sequential processing of the speech signal (left–right-analysis, island-driven strategy), but on the basis of structural relations. This means a new word hypothesis is accepted for a partial interpretation if the word hypothesis satisfies the expectation resulting from the semantic network and satisfied the restrictions of the constraint propagation process. As shown in Section 3, these restrictions are collected in modified concepts. The aim of the linguistic analysis is the instantiation of a concept representing a user dialog step. Due to the uncertainty of the word recognition module and due to the diverse forms of expression in spoken language neither a strictly data-driven nor a pure model-driven strategy seems to be promising. Therefore, we use a strategy working both on the acoustic data as well as on the expectations from the linguistic model [18].

For a goal directed search of the A^*-algorithm a judgment vector is used reflecting the compatibility, the quality, the reliability and the relevance of a partial interpretation. In more detail, the vector has the following components:

- *Structural compatibility*. This is a binary measure, which tests the linguistic restrictions, i.e., congruence of case, number and gender in a noun group.
- *Acoustic quality* of the underlying word or word chain hypotheses + **estimate** for the not covered speech signal: The acoustic score is generated by the EVAR word verification module and is the negative logarithmic probability of a continuous density Hidden Markov Model [51]. The estimate bases on statistical assumptions about the distribution of correct hypotheses.
- *Number of frames of the word chain with longest duration*. As the quality of word hypotheses reflects a distance-measure the following statement is valid for hypotheses with equal quality: hypotheses with longer duration are more probably correct hypotheses than shorter ones [50]. Therefore, this is a measure for the reliability.
- *Number of masked frames*. Measure of relevance, because the analysis goal can be reached in fewer steps.

The comparison between two interpretations is defined by the lexical order of their judgment vectors, i.e.

$$(x_1, \dots, x_k) < (y_1, \dots, y_k) \Leftrightarrow \exists x_i[x_i < y_i],$$
$$1 \leq i \leq k \wedge \forall x_l[x_l = y_l], \quad l < i.$$

This means first x_1 and y_1 are compared. If they are equal x_2 and y_2 are compared and so on. This is done until one component is greater than the corresponding one in the other vector. Moreover, for the second and third components of the vectors, only interval values and not the exact values are used. Full details of the judgments are given in [46].

Figure 35 shows the content of a search tree node after the complete linguistic interpretation of the first user utterance. By the network of instances, every spoken word is assigned to a syntactic, semantic and task-specific interpretation. The utterance 'ich möchte abends nach München fahren' ('I want to go to Munich this evening') is interpreted as a request for timetable information with destination 'München' ('Munich') and departure time 'abends' ('this evening'). As no departure place is detected the default 'Bielefeld' is used.

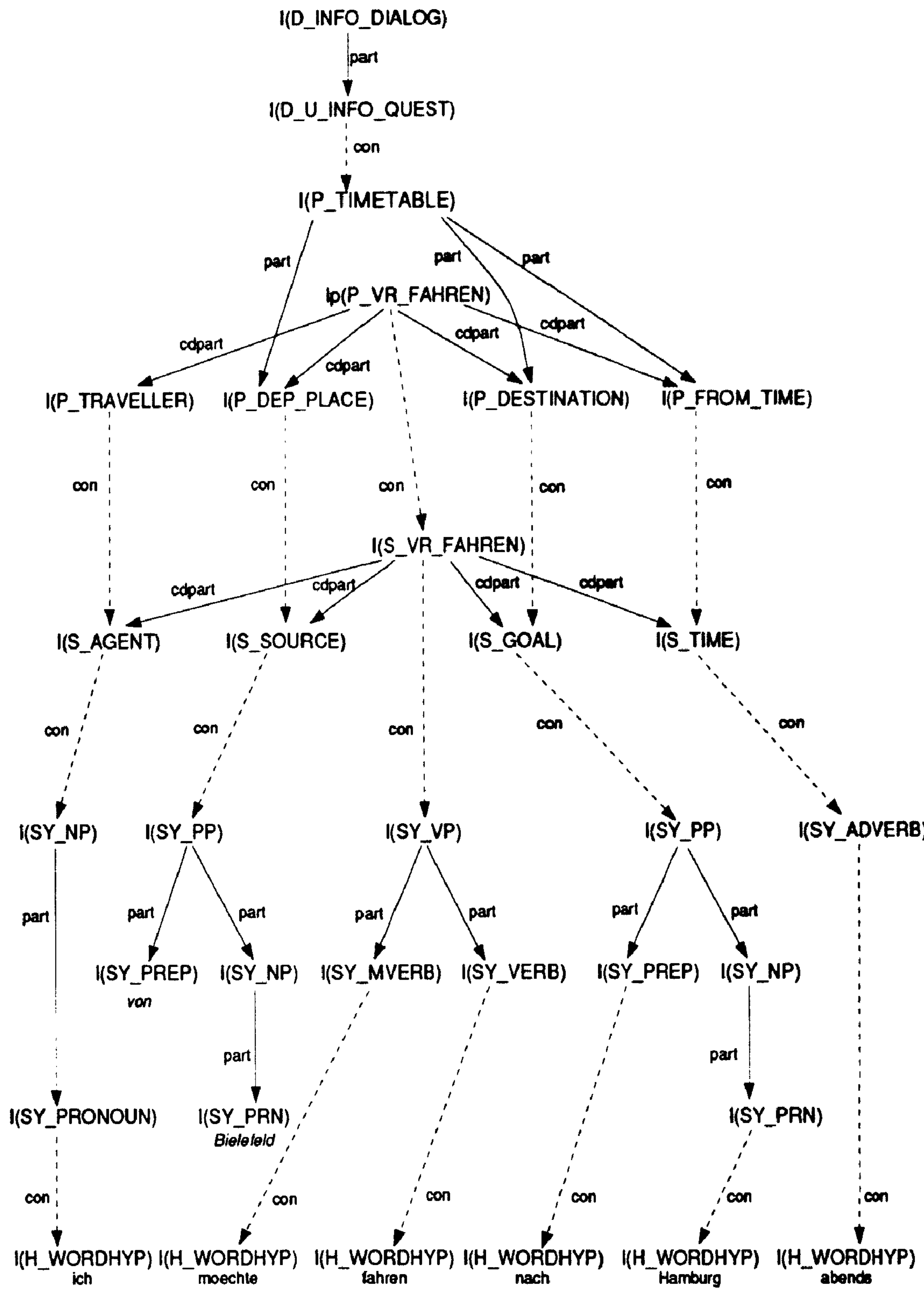


Fig. 35. Content of a search tree node with a complete interpretation.

The presented results are obtained under the following experimental framework:

- For the word recognition and verification task the ISADORA system [51] is used.
- No language model is used. Therefore, the perplexity is about the lexicon size of 1081 inflected forms.
- The acoustic module is trained with 500

domain specific sentences from four male speakers.
- The recognition module attains a word accuracy of 74.5% and a sentence recognition rate of 35%.
- The linguistic analysis works with the word hypotheses resulting from the 10 best-scored word chains of the acoustic module.
- For each dialog the test speaker speaks the first sentence of the dialog into the microphone and the analysis process begins. Due to the interpretation of the system a suitable answer (confirmation, correction, additional information) is given. This process is repeated until the dialog is successfully completed or failed.

Due to the recognition rate of the acoustic module, in many utterances not all of the spoken words are hypothesized for the linguistic analysis. To manage this problem the requirements for the coverage of the speech signal with word hypotheses are reduced to 2/3. This allows a correct interpretation of the utterance in spite of missing spoken words. Together with a dialog strategy which requests information missing for a data base inquiry by a check-back partial interpretations can be completed. Furthermore, incorrect interpretations can be corrected by a clarification dialog. During that dialog phase the user can confirm or correct all parts of the interpretation until the desired information is available.

One speaker of the training phase tested the system by 50 dialogs. 60% of the dialogs were completed successfully without an incorrect interpretation. 14% of the dialogs could be completed successfully after a clarification phase. For the remaining 26% no successful dialog was carried out. The main reason was the insufficient acoustic analysis hypothesizing less spoken words than needed for the coverage. Therefore, with an improved word recognition module the results could be improved.

6. Conclusions and further developments

The variability of the different applications, the achieved results and the efficiency of the solutions show that the viewpoint of knowledge-based signal understanding as an optimization problem lead to a powerful environment for a large class of pattern understanding problems. The goal is achieved by a general system shell which combines knowledge representation and use with a flexible control algorithm. This integration of different aspects and their adaptation to each other results in both flexibility and efficiency. Furthermore, the ERNEST shell does not force a special pardigm for the judgment of results. The restrictions on a special calculus are weak. The only required feature is the monotony concerning decreasing quality or increasing costs. As a consequence, the applications make use of judgment functions which are adequate for their fields of problems. For example, the medical ones are working with single fuzzy membership values, the industrial scene application uses fuzzy components measuring quality and priority, and last but not least the judgments in the EVAR system are based on probabilities calculated by hidden Markov models and heuristic measurements on the linguistic relevance of hypothesis. Because of the complexity of knowledge bases as well as the large number of nodes in a search tree, an explanation facility is of great importance for such environments. Depending on the application, it is designed for the system designer or for the user of a system. ELK provides general tools for both directions. Finally, it should be mentioned that the realization only uses general tools which are available in every UNIX-like operating system.

Our future work concerning knowledge-based signal understanding will be twofold. On the one hand further applications will be developed by using the ERNEST shell. Additionally, the system shell itself will be improved and extended. In order to guide the user by the selection of the best suited explanation units, options and parameters a knowledge-based user interface of ELK will be developed. ERNEST itself will be the shell to specify knowledge about the user and about explanation sequences. A first realization of a system component for automatic knowledge acquisition will be extended and improved. Other problems which are still unsolved concern the selfadaption of the control algorithm to prior runs on the same task, the integration of artificial neural networks in semantic networks, the parallelization of the system, and the development of time synchronous control algorithms for image sequences [55].

Acknowledgment

This work has been supported by the German Research Foundation (DFG).

References

[1] J. Bertin, *Graphische Semiologie*, De Gruyter, Berlin, 1974.

[2] J.C. Bezdek, *Pattern recognition with Fuzzy Objective Function Algorithms*, Plenum, New York, 1981.

[3] B.G. Buchanan and E.H. Shortliffe, eds., *Rule-Based Expert Systems*, Addison-Wesley, Reading, MA, 1984.

[4] B. Chandrasekaran, M.C. Tanner and J.R. Josephson, 'Explanation: The role of control strategies and deep models", in: J.A. Hendler, ed., *Expert Systems: The User Interface*, Ablex, Norwood, NJ, 1988, pp. 219–247.

[5] C.H. Chen, ed., *Pattern Recognition and Artificial Intelligence*, Academic Press, New York, 1976.

[6] W.J. Clancey and R. Letsinger, "NEOMYCIN: Reconfiguring a rule-based expert system for application of teaching", *Proc. 7th Internat. Joint Conf. on Artificial Intelligence*, Vancouver, BC, Canada, 1981, pp. 829–836.

[7] R. DeMori,*Computer Models of Speech Using Fuzzy Algorithms*, Plenum, New York, 1983.

[8] D. Dubois and H. Prade, Possibilistic logic, preferential models, non-monotonicity and related issues, *Proc. 12 Internat. Joint Conf. on Artificial Intelligence, IJCAI-91*, Darling Harbour, Sydney, Australia, 1991, pp. 419–424.

[9] C. Fillmore, "A case for case", in: E. Bach and R.T. Harms, eds., *Universals in Linguistic Theory*, Holt, Rinehart & Winston, New York, 1968, pp. 1–88.

[10] N.V. Findler, ed., *Associative Networks*, Academic Press, New York, 1979.

[11] G.W. Furnas, "Generalized fisheye views", in: M. Mantei and P. Orbeton, eds., *Proc. CHI 86 Human Factors in Computing Systems*, New York, 1986, pp. 16–23.

[12] A.R. Hanson and E.M. Riseman, eds., *Computer Vision Systems*, Academic Press, New York, 1978.

[13] G.E. Hinton, "Mapping part-whole hierarchies into connectionist networks", *Artificial Intelligence*, Vol. 46, 1990, pp. 47–75.

[14] H. Kitano and T. Higuchi, "Massively-parallel memory-based parsing", in: J. Mylopoulos and R. Reiter, eds., *Proc. 12 Internat. Conf. on Artificial Intelligence, IJCAI-91*, Darling Harbour, Sydney, Australia, 1991, pp. 918–924.

[15] A. Kobsa and W. Wahlster, eds., *User Models in Dialog Systems*, Springer, Berlin, 1989.

[16] R.A. Kowalski, "Predicate logic as a programming language", *Information Processing*, Vol. 74, 1974, pp. 569–574.

[17] D. Kumar, ed., *Current Trends in SNePS - Semantic Network Processing Systems*, Lecture Notes of AI, Vol. 437, Springer, Berlin, 1990.

[18] F. Kummert, Flexible Steuerung eines sprachverstehenden Systems mit homogener Wissensbasis, PhD thesis, Tecnische Fakultät der Universität Erlangen-Nürnberg, 1991.

[19] V.R. Lesser, R.D. Fennell, R.D. Erman and D.R. Reddy, "Organization of the HEARSAY II speech understanding system", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 23, 1975, pp. 11–24.

[20] B.T. Lowerre, The HARPY Speech Recognition System, PhD thesis, Dept. Comput. Sci., Carnegie-Mellon University, Pittsburgh, PA, 1976.

[21] R. McAleese and C. Green, eds., *Hypertext: State of the Art*, Intellect Limited, London, 1990.

[22] M. Minksky, "A framework for representing knowledge", in: P.H. Winston, ed., *The Psychology of Computer Vision*, McGraw-Hill, New York, 1975, pp. 211–277.

[23] J. Mylopoulos and H.J. Levesque, "An overview of knowledge representation", in: M. Brodie, J. Mylopoulos and J.V. Schmidt, eds., *On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases and Programming Languages*. Springer, New York, 1983.

[24] M. Nagao and T. Matsuyama, *A Structural Analysis of Complex Aerial Photographs*. Plenum, New York, 1980.

[25] H.-H. Nagel, "From image sequences towards conceptual descriptions", *Image and Vision Computing*, Vol. 6, 1988, pp. 59–74.

[26] A.M. Nazif and M.D. Levine, "Low level image segmentation: An expert system", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 6, 1984, pp. 555–577.

[27] A. Newell, "Production systems: Models of control structures", in: W.C. Chase, ed., *Visual Information Processing*, Academic Press, New York, 1973, pp. 463–526.

[28] H. Niemann, *Pattern Analysis*, Springer, Berlin, 1981.

[29] H. Niemann, *Pattern Analysis*, Springer, Berlin, 1981.

[30] H. Niemann, *Pattern Analysis and Understanding*, Springer Series in Information Sciences 4, Springer, Berlin, 2nd Edition, 1990.

[31] H. Niemann and H. Bunke, *Künstliche Intelligenz in Bild- und Sprachanalyse*, Teubner, Stuttgart, 1987.

[32] H. Niemann, H. Bunke, I. Hofmann, G. Sagerer, F. Wolf and H. Feistel, "A knowledge based system for analysis of gated blood pool studies", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 7, 1985, pp. 246–259.

[33] H. Niemann, H.Brünig, R. Salzbrunn and S. Schröder, "A knowledge-based vision system for industrial applications", *Machine Vision and Applications*, Vol. 3, 1990, pp. 201–229.

[34] H. Niemann, G. Sagerer, S. Schröder and F. Kummert, "Ernest: A semantic network system for pattern understanding", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 12, 1990, pp. 883-905.

[35] H. Niemann, G. Sagerer, U. Ehrlich, G. Schukat-Talamazzini and F. Kummert, "The interaction of word recognition and linguistic processing in speech understanding", in: P. Laface and R. DeMori, eds., *Speech Recognition and Understanding*, NATO ASI Series F 75, Springer, Berlin, 1992, pp. 425–453.

[36] H. Niemann, D. Wetzel, P. Weierich, G. Sagerer and K. Glückert, "Methods of artificial intelligence in medical imaging", in: *7. convegno internazionale e mostra sulle applicazioni della computer graphics nella produzione, progettazone e gestione (I.CO.GRAPHICS 1992)*, Mondadori Informatica SPS-AICOGRAPHICS, Milano, 1992, pp. 253-260.

[37] N.J. Nilsson, *Principles of Artificial Intelligence*, Springer, Berlin, 1982.

[38] N.J. Nilsson, *Principles of Artificial Intelligence*, Springer, Berlin, 1982.

[39] Y. Ohta, *Knowledge-Based Interpretation of Outdoor Natural Colour Scenes*, Pitman, Boston, 1985.

[40] S. Rubin, *The ARGOS Image Understanding System*, Dept. Comput. Sci., Carnegie-Mellon Univ., Pittsburgh, 1978.

[41] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing*, Volume 1: Foundations, MIT Press, Cambridge, MA, Eighth Edition, 1988.

[42] G. Sagerer, *Darstellung und Nutzung von Expertenwissen für ein Bildanalysesystem*, Volume 104 of *Informatik Fachberichte*, Springer, Berlin, 1985.

[43] G. Sagerer, *Darstellung und Nutzung von Expertenwissen für ein Bildanalysesystem*, Volume 104 of *Informatik-Fachberichte*, Springer, Berlin, 1985.

[44] G. Sagerer, "A multilevel parsing procedure based on dynamic programming for noisy input strings", in: I.T. Young et al., eds., *Signal Processing III: Theories and Applications*, Elsevier (North-Holland), Amsterdam, 1986, pp. 517-520.

[45] G. Sagerer, *Automatisches Verstehen gesprochener Sprache*, Volume 74 of *Reihe Informatik*, Bibliographisches Institut, Mannheim, 1990.

[46] G. Sagerer and F. Kummert, "Knowledge based systems for speech understanding", in: H. Niemann, M. Lang and G. Sagerer, eds., *Recent Advances in Speech Understanding and Dialog Systems*, NATO ASI Series F, Vol. 46, Springer, Berlin, 1988, pp. 421–458.

[47] G. Sagerer, U. Ehrlich, F. Kummert, H. Niemann and E.G. Schukat-Talamazzini, "A flexible control strategy with multilevel judgements for a knowledge based speech understanding system", in: *9th Internat. Conf. on Pattern Recognition*, Rome, 1988, pp. 788–790.

[48] G. Sagerer, R. Prechtel and H.-J. Blickle, "Ein System zur automatischen Analyse von Sequenzszintigrammen des Herzens", *Der Nuklearmediziner*, Vol. 3, 1990, pp. 137-154.

[49] H. Schicha and D. Emrich, *Nuklearmedizin in der kardiologischen Praxis: Methodem–Ergebniss–Indikation*, GIT-Verlag-Ernst Giebeler, Darmstadt, 1984.

[50] E.G. Schukat-Talamazzini, *Generierung von Worthypothesen in kontinuierlicher Sprache*, *Informatik-Fachberichte*, Vol. 141, Springer, Berlin, 1987.

[51] E.G. Schukat-Talamazzini and H. Niemann, Das ISADORA-System - Ein akustisch-phonetisches Netzwerk zur automatischen Spracherkennung, in: B. Radig, ed., *Mustererkennung 1991*, *Informatik Fachberichte*, Vol. 290, Springer, Berlin, 1991, pp. 251-258.

[52] A. Sohn and J.-L. Gaudiot, "Representation and processing of production systems in connectionist architectures", *Internat. J. Pattern Recognition Artificial Intell.*, Vol. 4, 1990, pp. 199-214.

[53] W.R. Swartout, "Knowledge needed for expert system explanation", *Future Computing Systems*, Vol. 1, No. 2, 1986, pp. 99-114.

[54] L. Tesniere, *Eléments de Syntaxe Structurale*, 2nd Edition, Klincksieck, Paris, 1966.

[55] C. Weighardt and H. Niemann, "Eine Inferenzkomponente für die Bildsequenzanalyse", in: B. Radig, ed., *Mustererkennung 1991*, Springer, Berlin, 1991, pp. 111-120.

[56] N. Woodhead, ed., *Hypertext & Hypermedia: Theory and Application*, Addison-Wesley, Reading, MA, 1991.

[57] L. Wos, R. Overbeek, E. Lusk and J. Boyle, *Automated Reasoning, Introducing and Applications*, Prentice Hall, Englewood Cliffs, NJ, 1984.

[58] M. Yachida and S. Tsuji, "A versatile machine vision system for complex industrial parts", *IEEE Trans. Comput.*, Vol. 26, 1977, pp. 882-894.

[59] J. Yen, "Generalizing term subsumption languages to fuzzy logic", *Proc. 12th Internat. Joint Conf. on Artificial Intelligence*, Darling Harbour, Sydney, Australia, 1991, pp. 472-477.

[60] L.A. Zadeh, 'Knowledge representation in fuzzy logic", *IEEE Trans. Knowledge Data Engrg.*, Vol. 1, 1989, pp. 89-100.