An Algorithm for Any–Time Speech Understanding *

V. Fischer, J. Fischer, H. Niemann
Lehrstuhl für Mustererkennung,
Universität Erlangen–Nürnberg
Martensstr. 3, D–91058 Erlangen, Germany
Phone: +49/9131/85-7775 Fax: +49/9131/303811
e-mail: fischer@informatik.uni-erlangen.de

To appear in
Proc. on the 3^{rd} German–Slovenian Workshop
on Speech and Image Analysis, Ljubljana, 1996

Abstract

This paper presents a parallel control algorithm for pattern understanding using a sematic network formalism for knowledge representation. Any-time capabilities, which are important for real world applications, are achieved by the use of iterative optimization techniques, like e.g. genetic algorithms, and the parallel processing of knowledge.

First results from an application of the algorithm in the linguistic analysis of the speech understanding system EVAR are given to demonstrate the feasibility of the approach.

1 Introduction

Knowledge based processing of complex patterns, like e.g. the understanding of continuously spoken speech, requires the mapping of sensor data into a system's internal model of the task domain in order to fulfill a given task in an optimal manner. Usually this mapping is provided by several different operations which may be divided into two phases of processing. First, in a data-driven phase operations for preprocessing and initial segmentation are applied to achieve a description of the signal in terms of meaningful simple constituents, like e.g. word hypotheses. Results are input to knowledge based or symbolic

*Part of this work was supported by the Real World Computing Partnership (RWCP).

processing, which has to extract the information relevant to perform a suitable reaction by man or machine. Due to both non perfect segmentation and ambiguous knowledge this requires the processing of many competing intermediate results, which is considered as a main reason for the fact that symbolic processing usually cannot meet real time restrictions imposed by many applications.

Parallel processing offers the desired speed, and a variety of algorithms for problems from data-driven processing have been developed, especially in image processing [8]. In contrast, parallel symbolic processing is much less investigated, although some major problems of the field, like e.g. parallel search techniques [9] or parallel knowledge representation [3], are discussed in the literature.

In this paper we describe a control algorithm for semantic network based pattern understanding and its use in a speech understanding system. The algorithm focusses on the achievement of any-time behaviour by both parallel knowledge processing and parallel iterative optimization. Section 2 briefly describes the network formalism and introduces an example which is used throughout the remainder of the paper. Section 3 gives the main idea of the control algorithm and Section 4 discusses the development of parallel inferences in the sematic network that allow both a fast computation of the desired symbolic description and the efficient restriction of large search spaces. Finally, Section 5 reports first results, and Section 6 gives a conclusion and an outline of further work.

2 Knowledge Representation

A semantic network is a directed, labeled graph consisting of nodes (called *concepts*) for the representation of facts or objects and links that provide relations between concepts. In our formalism, which is discussed in detail in [6], a concept may have an arbitrary number of *attributes* and structural *relations* for the definition of its (physical) properties and their dependencies.

Concepts may be related to another by three different types of links. Whereas *part* links for the decomposition of a concept into simple constituents can be found in most semantic network systems, *concrete* links are introduced to allow the representation of different levels of abstraction that must be considered in pattern analysis. Finally, *specialization* links which are also common to most semantic networks, are used to establish inheritance of attributes, relations, and links from general concepts to more special ones.

In order to increase the efficiency of knowledge representation, a concept is allowed to represent several variants of an object or a conception. For that purpose, part and concrete links may be marked *obligatory* or *optional* and may be grouped in *sets of modalities*, each modality providing a valid description of a concept. Note, that by this means less concepts are neccessary, and therefore the size of the knowledge base is kept moderate, but ambiguities are introduced that have to be resolved during analysis.

Figure 1 gives an example from the knowledge base of the speech understanding system EVAR [7]: On the morpho–syntactical level, a complex syntactic constituent, the noun

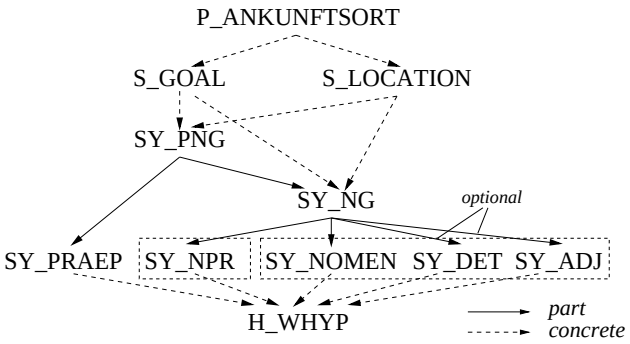

Figure 1: An excerpt from the linguistic knowledge base of the speech understanding system EVAR [7].

group (SY_NG), is decomposed into syntactic classes, like e.g. article (SY_DET), adjective (SY_ADJ), noun (SY_NOMEN), and proper noun (SY_NPR). As indicated by the dashed boxes, a noun group may consist either of a proper noun on its own (first modality of SY_NG, like e.g. "Erlangen") or may be build up from an article, an adjective and a noun (second modality, like e.g "the next train"). Since article and/or adjective could be omitted in an utterance these parts are marked optional in the latter. Via a concrete link each of those concepts is linked to a concept H_WHYP which serves as an interface to intial segmentation, i.e. word recognition.

The utilization of knowledge for the computation of *instances* and *modified concepts* is defined by six inference rules [6]. Instances establish a correspondence between some portion of the signal and the related concepts. The computation of an instance (e.g. for SY_PNG) is strictly data–driven and demands the existence of instances for at least all obligatory parts (e.g. SY_PRAEP, SY_NG) and concretes. In contrast, the computation of *modified concepts* for the use of restrictions from intermediate results may also be modell–driven, i.e. values for SY_NG may be restricted, if a modified concept for SY_PNG already exists.

3 An iterative control algorithm

In Section 1 the computation of a symbolic description that allows the optimal fulfilment of a given task was defined as the objective of knowledge based processing. If the goals of analysis are represented by competing *goal concepts* $C_{g_i}, 1 \leq i \leq \kappa$, in the semantic network formalism described above, the desired description can be obtained from an instance $I(C_{g_i})$.

Assuming the availability of a heuristic judgement function G for the scoring of instances as a prerequisite, instantiation of the goal concepts results in a judgement vector

$$\boldsymbol{g} \quad = \quad (G(I(C_{g_1})), \ldots, G(I(C_{g_i})), \ldots, G(I(C_{g_\kappa}))), \tag{1}$$

where $I(C_{g_i})$ is the instance computed for the i-th goal concept. An optimal description is provided by an instance with a maximal judgement $G(I(C_{g_i}))$, and it is the purpose of

a control algorithm to provide an efficient solution to this optimization problem. Utilizing the fact that instantiation of concepts only depends on

- the assignment $(A_i, O_j^{(i)})$, $i = 1, \ldots, \mu$, of segmentation results O_j to some attributes A_i of primitive concepts, and
- the choice $(C_k, H_l^{(k)})$, $k = 1, \ldots, \lambda$, of a modality H_l for each instance of a concept C_k that enables multiple definitions of an object,

the computation of a best scored instance may be solved by combinatorial optimization. For that purpose, the current *state of analysis* is summarized in a $(\mu + \lambda)$–dimensional state vector

$$\boldsymbol{r} \quad = \quad ((A_i, O_j^{(i)}); (C_k, H_l^{(k)})) \tag{2}$$

and the result of instantiation is rewritten as a function

$$\boldsymbol{g}(\boldsymbol{r}) \quad = \quad (G(I(C_{g_1})), \ldots, G(I(C_{g_i})), \ldots, G(I(C_{g_\kappa}))|\boldsymbol{r}), \tag{3}$$

of the state vector $\boldsymbol{r}$.

For the reliable computation of a best scored instance from the judgement vector in Equation (3) a cost function $\phi(\boldsymbol{r})$ is introduced. The distance

$$\phi(\boldsymbol{r}) = \min_i \{(\boldsymbol{e}_i - \boldsymbol{g}(\boldsymbol{r})/||\boldsymbol{g}(\boldsymbol{r})||)^2\} \tag{4}$$

from an ideal decision function $\boldsymbol{e}_i$ is an appropriate choice, provided that a perfect segmentation and the choice of the "correct" state of analysis support the instantiation of a single goal concept. For example consider the decision functions $\boldsymbol{e}_i$ defined by

$$\boldsymbol{e}_i \quad = \quad \begin{cases} 1 & \text{if} \quad C_{g_i} \text{ provides the desired symbolic description} \\ 0 & \text{else.} \end{cases} . \tag{5}$$

For the minimization of ϕ, iterative optimization procedures, like e.g. *threshold acceptance* and the *great deluge algorithm* [2], or *genetic algorithms* [5], are applied. Figure 2 gives an outline of a genetic algorithm for optimal instantiation, since in a comparative study [4] this algorithm results in the best speed of convergence.

Parallel iterative optimization is provided by a simultaneous evaluation of state vectors, or — in case of genetic algorithms — subsets of state vectors. To prevent a single search from convergence in a local minimum, processors may exchange results according to different strategies; see [1]. Since communication is expensive in our computing environment we employ a *simultaneous independent search*, i.e. do no state exchange until the first processors has finished its search.

The use of iterative optimization results in the desired any-time–behaviour, since a coarse solution is obtained, if less computing time is available, and a refined result is computed, if more iterations can be performed. However, according to Equation (3) the evaluation of a state vector $\boldsymbol{r}$ requires the computation of instances for the goal concepts in each iteration. Since a decrease in computing time for each iteration will result in a better any-time–behaviour, in the next section parallel processing is applied to the semantic network operations neccessary for the computation of instances.

create a set of state vectors $R_c := \{\boldsymbol{r}_{c,1}, \ldots, \boldsymbol{r}_{c,p}\}$
FOR ALL $\boldsymbol{r}_{c,i} \in R_c$
compute $\boldsymbol{g}(\boldsymbol{r}_{c,i})$ and $\phi_{c,i} = \phi(\boldsymbol{r}_{c,i})$ according to Equation (3) and (4).
WHILE computing time is still available
initialize a new set $R_n := \emptyset$
create a temporary set $R_t := \{\boldsymbol{r}_{t,1}, \ldots, \boldsymbol{r}_{t,p}\}$ by application of *selection*, *crossover* and *mutation* to elements from R_c
FOR ALL $\boldsymbol{r}_{t,i} \in R_t$
compute $\boldsymbol{g}(\boldsymbol{r}_{t,i})$ and $\phi_{t,i} = \phi(\boldsymbol{r}_{t,i})$ according to Equation (3) and (4).
select the p best states from $R_c \cup R_t$ and put them into R_n
let $R_c := R_n$ and report the best scored instance obtained from the state vector $\boldsymbol{r}_{c,best}$ with lowest cost as the solution

Figure 2: A genetic algorithm for optimal instantiation.

4 Parallel Knowledge Processing

Parallel semantic network systems usually make use of an isomorphic mapping of concepts of the knowledge base on the processors of a parallel hardware [3], which is appropriate if concepts are simple. To prevent concepts with a large number of attributes and relations from being a bottleneck in parallel instantiation, in our approach to parallel knowledge processing a network of concepts is automatically converted into a fine grained task graph. This is achieved by a two–stage process which first computes the number of (obligatory and optional) instances needed for the instantiation of goal concepts by a top–down expansion of the knowledge base, see Figure 3 for an example from the network in Figure 1.

The expanded network is then refined into a directed, acyclic graph $D = (V, E)$, where each node $v_i \in V$ represents the computation of an attribute, a relation, or the judgement of an instance. The creation of links requires the examination of dependencies between the procedures attached to the concepts. If node v_i is an argument to the procedure that computes a value for node v_j, a directed link $e_{ij} = (v_i, v_j) \in E$ is provided to express that computation of node v_i has to be finished before the computation of v_j may start.

For the computation of instances, nodes of the task graph are executed in parallel from bottom to top of the graph, starting with the attributes that provide an interface to segmentation results, and finishing with the judgement of goal concepts. Thus, instantiation is strictly data–driven, which is appropriate, if both results from segmentation and knowledge are quite unambiguous. However, in order to restrict the search space for iterative optimization, the combination with a modell–driven propagation of constraints from intermediate results is useful.

The construction of the task graph strongly supports the propagation of constraints, since it allows to use restrictions from all levels of knowledge both *once before* and *during*

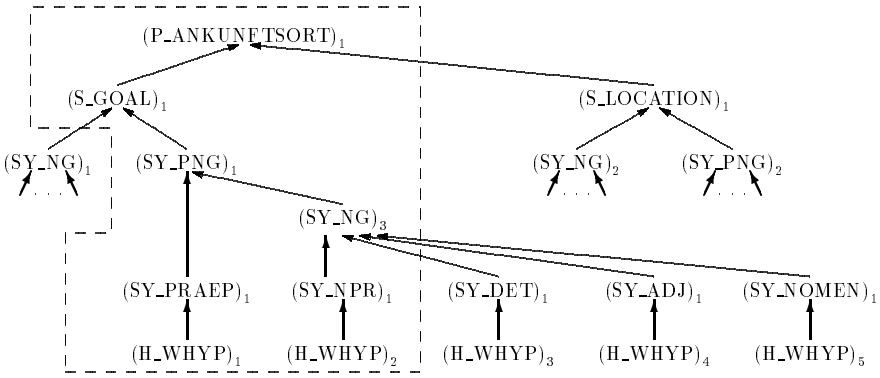

Figure 3: Excerpt from the expanded network for parallel instantiation.

analysis. As an example, consider the leftmost node (H_WHYP)$_1$ in Figure 3. Since from the construction of the expanded network it is apriori known that (H_WHYP)$_1$ is a concrete of the preposition (SY_PRAEP)$_1$, we can restrict the word hypotheses matching (H_WHYP)$_1$ to the set of prepositions. Because it is also known that (H_WHYP)$_1$ contributes to an instance of (P_ANKUNFTSORT)$_1$ (i.e. to a destination place), we can further restrict the set of all prepositions to the prepositions which are used to express this pragmatic intention (like e.g. "nach" ("to") in contrast to "von" ("from")). Moreover, if the destination has not yet been instantiated successfully in the n-th iteration of the control algorithm, we can restrict the task graph for iteration $n+1$ to the nodes associated with the instances in the dashed box. This way, computing time is saved by the computation of less nodes as well as by a reduction of the search space, since the modality for (SY_NG)$_3$ is restricted to the first one (i.e. the proper noun, cf. Section 2).

5 Experimental Results

As a testbed for our any–time approach to pattern understanding we use the linguistic analysis in the speech understanding system EVAR [7]. The aim of the system is to perform a dialog with a user in order to answer a request about the German InterCity train time table. For that purpose, knowledge about the syntax (concepts named SY_...., see Figure 1) and semantics (S_....) of the German language is represented in a semantic network as well as pragmatic knowledge (P_....). An additional level of abstraction (concepts named H_....) provides the interface to word recognition, and also a dialog model, which is not considered here, is integrated into the knowledge base.

The expansion of the knowledge base (154 concepts without dialog model) results in a network of approx. 6500 nodes, which is reduced to a network of approx. 1300 modified concepts by the modell-driven propagation of constraints from pragmatics (cf. Section 4). Thus, compared to the mere data-driven approach in [4] computing time for each iteration is decreased by a factor of 5. Moreover, due to the elimination of nodes, the search space for iterative optimization is reduced by several orders of magnitude. The final refinement

n	correct pragmatic intentions [%]				
	$p=1$	$p=2$	$p=3$	$p=4$	$p=5$
1	72.4	74.6	78.6	80.6	81.8
5	73.1	78.7	81.3	82.7	83.3
10	73.8	79.8	81.6	83.4	84.2
25	76.5	81.4	83.2	83.6	84.5
50	78.8	83.5	84.8	85.8	86.0

Figure 4: Percentage of correctly analyzed pragmatic intentions for n iterations and simultaneous independent search using $p = 1, \ldots, 5$ processors.

of modified concepts yields a task graph of approx. 11.000 nodes that have to be computed in each iteration of the algorithm.

In a first series of experiments we used 134 typical user requests in a natural language input mode, which, however, does not mean to exclude the assignments (A_j, O_i) from (word) hypotheses to initial nodes of the task graph from optimization. For example, the utterance "Ich möchte von *Erlangen* nach *Heidelberg* fahren." ("I want to go from *Erlangen* to *Heidelberg*") contains two proper nouns which may be assigned to $(\text{H\_WHYP})_2$ in Figure 3. Since in natural language input mode no acoustic score of a (partial) interpretation is provided, we use the number of frames of the word chain with longest duration for the judgement of instances. A new state of analysis is accepted, whenever the number of covered frames is increased.

A *pragmatic intention* is a concept of the task domain that defines a conception the system is able to talk about. Figure 4 reports the percentage of correctly classified pragmatic intentions with respect to both the number of iterations n and the number of processors p for parallel iterative optimization. Any-time characteristics are shown by the increasing percentage of correctly classified concepts with an increasing number of iterations. The most important pragmatic intention, i.e. the destination place, is correctly classified in 85 percent within $n = 5$ iterations ($p = 1$), and therefore the system should be able to start a clarification query about the missing intentions, like e.g. departure time. For $p = 2, \ldots, 5$ the benefits of parallel processing for an improvement of results becomes evident, but a more efficient parallel search has to be subject of further work.

6 Conclusion

In this paper a parallel control algorithm for semantic network based pattern understanding was applied to a speech understanding system.

Future work will concentrate on the achievement of dialog capabilities, and on the further improvement of iterative optimization by a dialog guided initialization of the search space. For the further improvement of real–time behaviour, incremental processing

of word hypotheses should be investigated.

References

[1] R. Azencott, editor. *Simulated Annealing. Parallelization Techniques.* Wiley–Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, Chichester, 1992.

[2] G. Dueck. New optimization heuristics: The great deluge algorithm and the record-to-record-travel. *Journal of Computational Physics*, 104(1):86–92, 1993.

[3] M. Evett, J. Hendler, and L. Spector. Parallel knowledge representation on the connection machine. *Journal of Parallel and Distributed Computing*, 22(2):168–184, 1994.

[4] V. Fischer. A Parallel Any–Time Control Algorithm for Image Understanding. submitted to: 13th Int. Conf. on Pattern Recognition, Vienna, 1996.

[5] D. Goldberg. *Genetic Algorithms: Search, Optimization and Machine Learning.* Addison–Wesley Publ. Co., Reading, Mass., 1989.

[6] F. Kummert, H. Niemann, R. Prechtel, and G. Sagerer. Control and explanation in a signal understanding environment. *Signal Processing*, 32(1-2, Special Issue on Intelligent Systems for Signal and Image Understanding):111–145, 1993.

[7] M. Mast, F. Kummert, U. Ehrlich, G. Fink, T. Kuhn, H. Niemann, and G. Sagerer. A speech understanding and dialog system with a homogeneous linguistic knowledge base. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(2):179–194, 1994.

[8] I. Pitas, editor. *Parallel Algorithms for Digital Image Processing, Computer Vision and Neural Networks.* Wiley Series in Parallel Computing. John Wiley & Sons, Chichester, 1993.

[9] B.W. Wah, G. Li, and C.F. Yu. Multiprocessing of combinatorial search problems. *IEEE Computer*, 18(6):93–108, 1985.