IDIAP COMMUNICATION

IDIAP

Martigny - Valais - Suisse

Towards a Multi-agents Approach for Understanding Speech

Murielle Vial Jean-Luc Cochard

IDIAP–Com 96-05

December 1996

Dalle Molle Institute for Perceptive Artificial Intelligence • P.O.Box 592 • Martigny • Valais • Switzerland

phone +41 – 27 – 721 77 11
fax +41 – 27 – 721 77 12
e-mail secretariat@idiap.ch
internet http://www.idiap.ch

Abstract

This paper describes a currently running project at IDIAP, aiming at the development of a continuous spoken language recognition system. A particular attention is paid to two important topics under investigation. Firstly, the system characteristics, namely its structure as a multi-agent system and secondly, its search mechanism of a global solution as an optimisation procedure, are studied. As this project is definitely an undergoing research activity, many still important open questions have to be addressed. The major one, concerning a mathematical foundation for the resolution mechanism is covered in a section dedicated to further developments.

1 Introduction

Continuous spoken language understanding (CSLU) is a very difficult task for which no satisfactory solution has yet been proposed. One particularity of CSLU is that it lies at the intersection of two distinct domains: signal processing and computational linguistics. Many innovative improvements are described daily to extend either coverage or performance of current solutions to very limited subproblems. These developments have very few impact on the concrete realizations of entire CSLU systems. One important reason has been identified for this state of things: as a major concern in the development of entire CSLU systems is to combine different components encoding knowledge at different levels of abstraction in a coherent and mathematically-sounded way. As low level processing (phonemes identification) is best implemented by Hidden Markov Models (HMMs), it has as a consequence that subsequent treatments have to be probabilistic in order to be easily and efficiently combined all together.

The description of a natural language by means of *probabilistic language models* as given by n-grams, is, from our point of view, a very useful but also very incomplete descriptive approach. Standard NLP — rule-based approaches — give rise to more complex and far more precise descriptions of language structures, but they lack in well covering large corpora.

Typical information of spoken language such as the ones conveyed by prosody or "liaison" phenomena occuring in French are clearly context sensitive. Their appearance in a speech utterance is constrained by rules those complexity lies in the variety of abstraction levels they use to be described. If, for example, the "liaison" (insertion of a /z/ sound between a word ending by a mute "s" and a word starting by a vowel) occurs at the phonemic level, it is connected to words orthography and it is only valid in some precise syntactic structures.

The need for an overall architecture that favours the manipulation of heterogeneous knowledge sources and that is flexible enough to take advantage of clues of any level as soon as it is sensible are the two major objectives of our research.

This paper will first describe a prototype system that addresses these two main concerns. This description is then followed by a discussion of two distinct approaches that have many common points with the $\mathsf{ETC}_{\mathsf{vérif}}$ approach.

2 $\mathsf{ETC}_{\mathsf{vérif}}$, a First Experiment

The system $\mathsf{ETC}_{\mathsf{vérif}}$ [CO95, CF95] under development at IDIAP, is implemented as a multi-agent system, and is based on a general purpose platform called ETC, for cooperative treatment environment ("Environnement de Traitement Coopératif"). The purpose of decomposing a CSLR[1] system into two layers: a kernel that is application-independent and a periphery that bears all the knowledge of the application domains, improves the design of a flexible and adjustable system. The ETC platform provides different services to the host application:

[1] Continuous Spoken Language Recognition.

- a model of Ω-*agents* that defines a standard interface to distinct entities representing specific areas of expertise;
- a model of μ-*agents* that gives rise to the notion of active data, allowing a decentralized structuring of local hypotheses;
- a framework for the definition of the system evolution dynamics.

$\mathsf{ETC}_{\mathsf{vérif}}$ is addressing a simplified problem of CSLR, namely verification of speech utterances. This concretely means that the input data of the system is twofold: the signal sample, on one side, and the text that had to be uttered, on the other side. This simplified context greatly reduces the required number of agents and the internal complexity of some agents that have to be developed. Nevertheless, many experiments can be conducted in this case that give us some valuable information on how to setup an entire and efficient CSLR system.

2.1 The System Architecture

The prototype $\mathsf{ETC}_{\mathsf{vérif}}$ is a multi-agent system based on ideas proposed by S. Lander in her PhD thesis [Lan94]. The decomposition of the world of agents into two very distinct classes, as explained below, is a specificity of our system (see also [CO95]).

Ω-agent, provider of solutions

The Ω-*agents* are problem solution providers. They have two capabilities:

- *solution initiation*: the agent competence is required to generate a solution hypothesis that solves a local sub-problem and that can be used as part of a composite solution.
- *solution adaptation*: the agent competence is used to provide an alternative solution with a focus of attention on what was conflicting in the previous proposal(s).

The Ω-agents currently used in the system, are HMMs and a GTP. The GTP agent produces a sequence of phonemes from a sentence in ASCII codes. Two HMM agents are available, one providing phonemes and the other providing words hypotheses. This last agent is bound to the GTP agent: its input is a speech sample, an intermediary output is a sequence of word labels; this sequence is the input of the GTP agent and the final result is a phonetic transcription of the words sequence. However, in a new release of our system, both phonemes and words hypotheses will cooperate together in the determination of an acceptable solution.

μ-agent, element of a composite solution

The μ-*agents* are the results provided by Ω-agents. They are small pieces of software embedding sub-problem solutions, that can be phonemes hypotheses, words hypotheses, prosodic labels, phonemes classes, etc. In the current version, only phonetic hypotheses are stored by μ-agents. Three distinct data items are stored by any μ-agent:

1. *Borne_inf*, the starting time, in the speech sample, this μ-agent refers to.
2. *Borne_sup*, the corresponding ending time in the speech sample.
3. *Id*, the identifier of the μ-agent (for example, a phonetic label).

Resolution Sketch

Figure 1 gives an overview ot the $\mathsf{ETC}_{\mathsf{vérif}}$ architecture. μ-agents initiate the resolution process by providing a first solution for the ones in charge of low level resolution. Each constituent of the solution gives rise to the generation of a μ-agent. The current set of μ-agents computes what is called a *reliability estimate* (see section 2.2.2). The *evolution engine* carries out a selection of the current best solution (see section 2.2.3). This best solution is processed in order to find islands of unreliable results —areas where the reliability of μ-agents are too low (see section 2.2.4). These unreliable results constitute the focus of attention for solution adaptation.

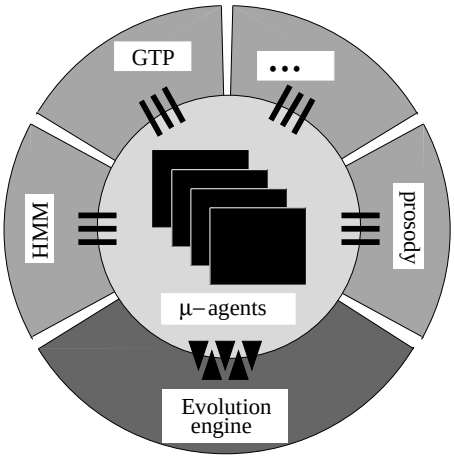

Figure 1: Architecture of $\mathsf{ETC}_{\mathsf{vérif}}$.

Implementation Concerns

The programs, implementing Ω-agents, are written in C and C-shell. However, they adopt a standard and uniform interface written in Oz [HSW93, HSW92]. Thus, Ω-agents can be considered as objects holding methods such as `bestSolution` and `anotherSolution` which allow them to provide, respectively, a first solution and to give another solution (or a solution different from the initial one) in a certain focus of attention.

$\mathsf{ETC}_{\mathsf{vérif}}$ is enriched by a graphical interface (see Figure 2). Every Ω-agent interface method can be started from a menu part of the graphical interface. For example, if one wants the HMM Ω-agent to initiate a solution, he has to choose `HMM bestSolution` in the menu `Init Solution` or, if he wants the HMMs to give another solution, he has to select a time interval then choose `HMM next` in the menu `Next Solution`. Practically, the user doesn't need to intervene in the resolution process because the selection of islands of unreliable results and their further process is automatically done. Nevertheless, the user has the possibility to participate freely at the resolution process by adding new partial solutions.

Once Ω-agents have been activated, their results encapsulated in μ-agents, appear graphically as clickable rectangles (see figure 2). This feature promotes the graphical interface as a debugging tool extremely useful in the current context of this system development.

2.2 Towards an Application-independent Platform

This section intends to explain some of the technical aspects of $\mathsf{ETC}_{\mathsf{vérif}}$ processing steps. A more detailed version of this part can be found in [CV95]. The main objective in the $\mathsf{ETC}_{\mathsf{vérif}}$ project is to provide an application-independent platform ETC, on which application-dependent modules are connected in order to supply the system with knowledge of a particular domain.

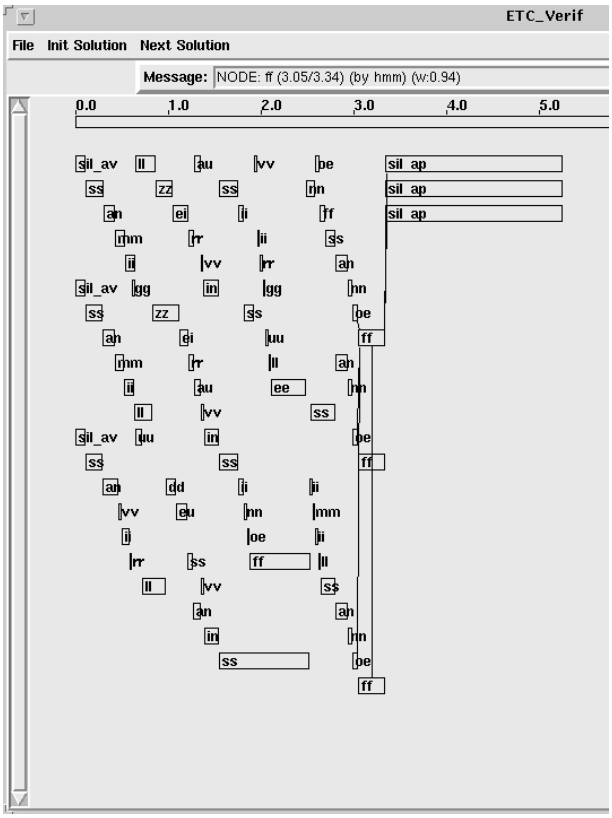

Figure 2: Illustration of the $\mathsf{ETC}_{\mathsf{vérif}}$graphical interface presenting phonetic hypotheses.

This means that the resolution strategy is dedicated to ETC. It is performed by a conditional loop, referred to as the *evolution engine* in Figure 1 (see also section 2.2.4). The distinct steps explained in details below, give rise to a dynamic system whose resolution strategy is event-driven like any conventional graphical interface, for example. The major difference is that events in the context of $\mathsf{ETC}_{\mathsf{vérif}}$, are hypotheses of sub-problems resolution provided by Ω-agents.

Concurrency and lasy programming are two major properties of Oz that are intensively used in the implementation of ETC, as will be clearly seen in the examples below.

2.2.1 Solution Initiation

Ω-agents are entities waiting for specific input. Once this input is available, they can provide the system with an initial solution. With regard to the decomposition of one solution in a set of μ-agents, there is no definite approach for the moment. As mentioned above, this first version of $\mathsf{ETC}_{\mathsf{vérif}}$ is dealing with phonemes and in a short future with word hypotheses and prosodic labels. It seemed clear to us that a phonetic label including temporal boundaries has to be considered as an independent entity able to determine how much it is reliable based on its context. This qualifies such a unit to be considered as a μ-agent with its own behaviour.

When a solution hypothesis is provided, this solution is split in a sequence of μ-agents. The protocol for the generation of a μ-agent looks like what is described in Figure 3. As soon as A is generated, it reports its presence to an object called `Broadcaster` (action 1). This object knows the list of all μ-agents[2] of the same class present in the system. This knowledge allows `Broadcaster` to send a message (action 2), containing some information about the newly generated μ-agent, such as

[2] μ-agents are referred to as `nodes` in listed source code.

its `id` and temporal boundaries to all the other μ-agents (denoted by B, in Figure 3), which compare this set of information to their own characteristics so as to establish or not relations with A (action 3).

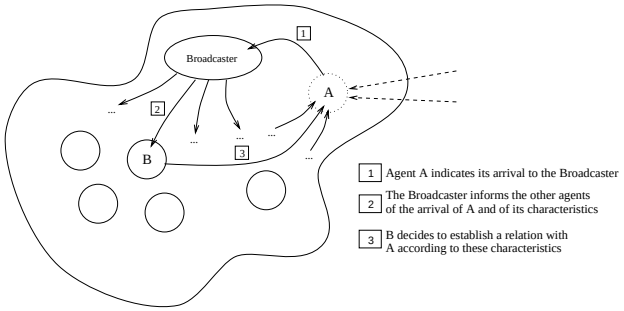

Figure 3: Operations done by a μ-agent when it is generated.

Three types of relation can be established between μ-agents containing phonetic information:

A precedes B: *upper boundary* of μ-agent A is almost equal to *lower boundary* of B.

A follows B: *lower boundary* of A is almost equal to *upper boundary* of B.

A is brother of B: *lower boundary* of A is almost equal to *lower boundary* of B, *upper boundary* of A is almost equal to *upper boundary* of B and their *ids* are compatible (same phonetic class).

So, if one of these properties is true, a relation is established between two μ-agents and they become *neighbours*: they store each other's reference according to their type of relation.

Information on nodes and their relations can been displayed by the graphical interface (see figure 2) by clicking on them: in our example, the node selected is the phoneme `ff`, placed in the time interval [3.05/3.34] secondes, provided by the HMM Ω-agent, with a reliability measure equal to 0.94 (see window *message*). The list of its neighgours is denoted by lines as illustred in the same figure.

Thus, Oz allows μ-agents to cooperate by exchange of messages allowing dynamic creation of links.

2.2.2 Computation of a Reliability Measure

The relations, as presented in section 2.2.1, are used to calculate a reliability measure[3] of each μ-agent. This measure is dynamic and evolves with generation of new μ-agents and creation of relations. The current measure is proportional to the number of neighbours, and to the reliability of each neighbour.

This principle leads to the following formulas:

$$
\begin{aligned}
R(A) &= \lim_{i \to \infty} R_i(A) \\
R_i(A) &= R_{i-1}(A) - \frac{1}{(2 + Rn_{i-1}(A))^i} \\
R_0(A) &= 1 \\
Rn_i(A) &= \sum_{B=1}^{N} \mathcal{N}(A,B) R_i(B)
\end{aligned}
$$

The reliability of A, denoted by $R(A)$, is defined as the value for which $R_i(A)$ converges. Practically, the series is computed as long as the improvement is greater than a Δ.

[3] The notion of *reliability measure* is referred to as a `Weight` in listed source code.

$$R_{i-1}(A) - R_i(A) < \Delta, \text{with } \Delta = 0.01$$

This value of Δ has been selected to ensure fast convergence. Moreover, since $Rn_i(A) > 0$, $R_i(A)$ converges in the interval $]0, 1[$.

$Rn_i(A)$ denotes the accumulation of reliability of A's neighbourhood. If the set of the μ-agents is denoted, at a given time, by the integers from 1 to N; $\mathcal{N}(A, B) = 0$ if A and B have no relation, else $\mathcal{N}(A, B) = 1$ for any kind of relation between A and B.

The computation of this reliability measure across a number of independent entities is a perfect illustration of the power of Oz with regards to distribution and synchronisation. The reliability measure in its current definition, imposes each round of current solution evaluation to start from an initial uniform value of `1.0` (upper-bound limit of the interval of possible values).

2.2.3 Optimal Path Selection

The optimal path is the sequence of μ-agents covering the entire problem space with the highest possible weights. Here again, there is no "super"-agent responsible of this task; this processing is distributed across the set of existent μ-agents at a particular time.Each μ-agent searches, among its preceding ones, which one is on the optimal subpath (see figure 4).

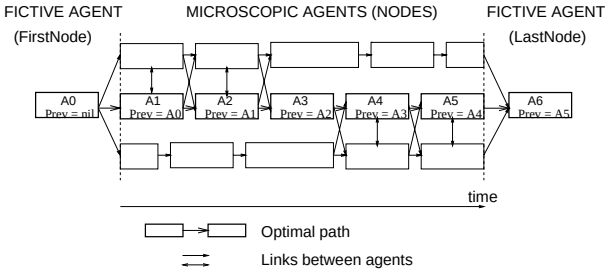

Figure 4: Graphical illustration of the best path selection.

The simplest way, to perform this selection, is to choose the path for which the sum of the weights is the highest. However, this solution has the drawback to favour long paths with low reliable μ-agents instead of short paths with high reliable μ-agents, as illustrated in Figure 5.

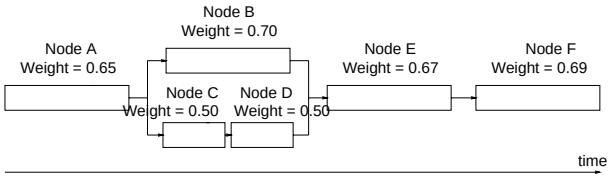

Figure 5: Node E prefers the path built of nodes A, C and D with a the sum of their weights is equal to 1.65, instead the path A, B with a sum equal to 1.35.

Another solution is to take the means of weights instead of the sum but this solution has another drawback, as demonstrated in Figure 6.

Finally, the selected approach computes a weight pondered by the time interval duration coded by the corresponding μ-agent. For the artifical examples presented in Figures 5 and 6, if we consider that large rectangles are representing solution elements with a double duration than narrow ones, the paths A, B, E, F and A, C, D, E, F are respectively selected as expected.

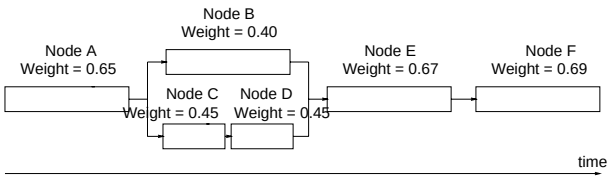

Figure 6: Node E chooses node B as preceding one on the optimal path because the means of weights computed for A and B is equal to 0.525, that is higher than the one of nodes A, C and D (equal to 0.517).

Determining the best path is an easy task: every μ-agent is requested to integrate information on the best possible path up to itself with its current weight, in order to provide information on the best path including itself.

As a side-effect of this computation, each μ-agent memorizes which is its preceding μ-agent on the best path. This means that the agent at the last position is able to give access to the entire best path.

2.2.4 Evolution Engine

Once the computation of reliability is finished and once every μ-agent has determined its best preceding neighbour, it is possible to make an iteration in the evolution process. The system has to focus on the best path in order to evaluate if the current best solution can be considered as an acceptable solution to the problem set down, or what are its weaknesses that makes it inacceptable.

The function of the *evolution engine* is to isolate the set of μ-agents which are part of the best path and for which the reliability measure is insufficiently high. Connected sequences of such unreliable μ-agents are called *islands of unreliability*. The set of these unreliable μ-agents is computed by the following program.

The selection of unreliable μ-agents is still an open question for which some alternatives are currently compared. The current solution is based on an average value of the reliability along the best path. The weak hypotheses are the ones that are at a certain distance below this average value.

The second step of the iteration consists in requesting additional solutions to the Ω-agents available in the system on these local problematic areas of the current solution. The generation of new hypotheses coded as μ-agents will give rise to a new iteration of evaluation-selection-decision, as explained above.

3 Discussion and Further Developments

This section gives an overview of two interesting approaches, namely the "whiteboard" approach and Harmony Theory, that have many interesting common points with principles implemented in $\mathsf{ETC}_{\mathsf{vérif}}$.

3.1 The "Whiteboard" Approach

The resolution strategy based on a *whiteboard* was introduced by Boitet and Seligman [BS94, BS95]. It has some interesting similarities with the structure of Ω- and μ-agents introduced in $\mathsf{ETC}_{\mathsf{vérif}}$. The *components* in [BS95] definitely play the same role than the one dedicated to Ω-agents. They are knowledge sources of different types, they encapsulate their own resolution mechanism and they are accessed through a standard interface, called *manager* in the whiteboard terminology.

The whiteboard central data structure is a huge graph manipulated by a *coordinator*. In $\mathsf{ETC}_{\mathsf{vérif}}$, there is no central and unique structure; it is replaced by a collection of inpedendent entities, the

μ-agents. Both architectures seem to play the same role, that is to centralize the different hypotheses that constitute a possible solution to the given problem.

Two important conceptual notions are also part of the objectives of both approaches:

1. The integration of heterogeneous knowledge sources. This is a strong claim as we consider important to provide a resolution framework able to deal with different types of knowledge. The underlying idea is to fill the gap between rule-based and statistical knowledge types.

2. The interleaving of layers reaction. In the whiteboard terminology, this feature is called *incremental processing*. In $\mathsf{ETC}_{\mathsf{vérif}}$, this feature is not explicitely mentioned. Nevertheless, as it is directly supported by the programming language used to implement our system, namely Oz, it is an important feature, we try to take advantage of.

The two design issues given above have in common that there exists plenty of good common-sense reasons to justify to incorporate as many knowledge sources as possible in a system and to let a particular constraint to apply as soon as possible. But implementation of this system is a long-term and complex task that can only be successfully achieved if based on strong mathematical foundations for distributed and concurrent problem solving.

3.2 Harmony Theory, the Missing Foundation

Harmony Theory (HT) has been described by P. Smolensky, in 1986, in [Smo86]. Since that time, no major use of this mathematical formalism has been carried out. Only some loose links can be seen in papers referring to the original description of HT.

Harmony Theory (HT), as it is meant in its original description, offers an alternative paradigm for computer science. Even if $\mathsf{ETC}_{\mathsf{vérif}}$ is not as much ambitious as HT, our research project tries nevertheless to define a new mechanism for the resolution of complex problems based on a multi-agents approach close to connectionist models. The current lack of mathematical foundation for $\mathsf{ETC}_{\mathsf{vérif}}$ has been identified as very harmful to further evolutions. Even if all the technical distribution and synchronization problems can be considered as solved, the current behaviour of $\mathsf{ETC}_{\mathsf{vérif}}$ is far from being satisfactory, when a speech utterance is submitted to its resolution strategy.

The current strategy of $\mathsf{ETC}_{\mathsf{vérif}}$ can be summarized as follows:

Let us consider P to be one speech utterance to be identified, and $\mathcal{D}$, the solution space of P, which is a cartesian product of all sets of possible hypotheses one can assign to any part of P. The strongest assumption in $\mathsf{ETC}_{\mathsf{vérif}}$ is that an optimal solution $s(P) \in \mathcal{D}$, can be found starting from a sub-optimal solution $s_0(P)$ to which a sequence of refinements is applied, giving rise to the list $s_1(P), s_2(P), \ldots s_n(P)$, with $s_n(P) \equiv s(P)$, of solutions satisfying incrementally more and more constraints imposed by knowledge sources.

In order to generate the dynamics of computing this sequence of solutions with as a goal to reach the target $s(P)$, a measure of intrinsec quality of every $s_i(P)$ is computed. This is performed by the reliability measure described in section 2.2.2. As it is implemented, this measure permits to identify parts of a solution $s_i(P)$ that are less accurate than others. This particularity is used to select hypotheses that need to be revised by asking knowledge sources to provide other hypotheses.

The paradigm of HT is surprisingly close to the principles described above, even if $\mathsf{ETC}_{\mathsf{vérif}}$ has been designed without any knowledge of this work. Two major differences have to be highlighted here in order to understand the strenghs of HT compared to our system.

The two-levels graph

HT assumes that a solution $s_i(P)$ — sub-optimal or optimal — can be described as a *two levels graph*. On one level, hypotheses of distinct classes can be stored. They are called *representational features* in the terminology of HT and they correspond to our μ-agents or the "white nodes" in the whiteboard approach. The second level is constituted of *knowledge atoms* for which there is no strict equivalent in $\mathsf{ETC}_{\mathsf{vérif}}$. In the latter, each μ-agent of a given class has its proper rules that determines its possible

relations with μ-agents of other classes. This technics clearly lacks in uniformity. If compared to the whiteboard approach, "grey nodes" are similar to knowledge atoms.

Nodes of one level are never related to nodes of the same level. The only possible relations are between nodes from one level and nodes from the other. One node can have multiple relations with the other level. This particularity can have two interpretations. If it is a knowledge atom that has multiple relations, this means that a conjonction of hypotheses have to be present to validate this item of knowledge. If it is a representational feature (an hypothesis) that has multiple relations, this means that this particular hypothesis is useful to many knowledge atoms.

The Harmony of a graph

On top of this graph structure, HT defines a measure of *Harmony*, just as the reliability measure is defined on top of μ-agents. But Harmony is far more sophisticated than reliability since it includes likelihood as a possible raw parameter assigned to a knowledge atom. It is out of the scope of this paper to give a precise and mathematical description of Harmony. If we refer to the strategy of $\mathsf{ETC}_{\mathsf{vérif}}$, summarized above, a resolution process based on HT has a very similar structure:

A sequence of states — a state corresponding to a given two-levels graph — is computed attempting to modify the global Harmony. This modification can be a decreasing value, in certain circumstances in order to jump out of local optimums. But ultimately, this modification tends to be an improvement.

The strongest and most limitating assumption of HT lies in the method to build a state out of its predecessor, that is how does the system evolve in the solution space. HT assumes that, in order to be tractable by a computer, the set of all representational features and the set of knowledge atoms are finite. This is definitely not the case in a spoken language understanding task. With this limitative restriction, nodes of the graph can be either active (they take part to the computation of the graph Harmony) or inactive. The evolution simply means to slightly change the set of active and inactive nodes.

Nevertheless, we have the strong intuition that there exists a possible combination of μ-agents of $\mathsf{ETC}_{\mathsf{vérif}}$ that constitute a finite subset of the infinite set of possible hypotheses virtually available through requests to Ω-agents, with an harmony-based evolution strategy. This combination will be the objective our next step towards the development of a multi-agents spoken language understanding system.

4 Conclusion

This paper gave an overview of a multi-agents architecture addressing a simplified problem of speech understanding. After having described in some details the structure and the resolution mechanism embedded in $\mathsf{ETC}_{\mathsf{vérif}}$, a discussion took place highlighting the problems faced by the current implementation. Two different approaches with many similarities with our system have been described. They definitely show that this multi-agents approach has to be further investigated before any define conclusion can be raised on its usefulness to solve very complex problems.

Nevertheless, concerning $\mathsf{ETC}_{\mathsf{vérif}}$, we are convinced that the current system is open to many extensions in the following directions, as it is clear that no commitment has been undertaken on one particular approach, in any of them:

Heterogeneity of knowledge sources – Up to now, only phonetic labels have been manipulated and implemented as active data structures (μ-agents). The extension towards other data types like word labels and prosodic labels is currently under consideration and nothing fundamental stands in the way of their integration in the system. Once this will be done, μ-agents won't only have connection inside their own class, just as they have now, but also between classes. For example, a sequence of phonemes that fits part of a theoretical phonetic transcription of a word will be connected each others. We are convinced that a probable solution to the problem

of automatic speech recognition needs an extensive set of knowledge sources with fine tuned connections between partial solutions they provide.

Distributed processing – In this current experience with Oz, it has been considered as a very profitable feature to be able to think in terms of concurrent entities with synchronism mechanims based on logic variables and don't care indeterminism. We consider that a lot of experience has to be gained in this paradigm and many improvements can probably be carried out in the kernel part of our system, namely ETC.

Resolution strategies – This system is addressing a crutial point of hard problems resolution: how to find its way out in a huge solution space without too much useless processing? Usual approaches such as "depth-first search" or "N-best search" cannot be considered as intellectually satisfactory solutions. It seems clear to us that a sub-optimal solution has high confident parts and low confident ones on which further processing is required.

Interactivity – The integration of Oz with a graphical interface toolkit is the best possible because Oz can be considered as an event-driven language. In this context, the end-user can easily be considered as a conventional Ω-agent providing part of solution to the system.

Acknowledgements

We are very grateful to our colleague Eddy Mayoraz who was able to discover interesting and valuable relationships between the approach implemented in $\mathsf{ETC}_{\mathsf{vérif}}$ and his memory of Harmony Theory, studied some years ago.

References

[Baj93] Ruzena Bajcsy, editor. *IJCAI 93, 13th International Joint Conference on Artificial Intelligence*, Chambéry, France, August 1993. Morgan Kaufmann Publ., Inc.

[BS94] C. Boitet and M. Seligman. The "Whiteboard" architecture: a way do integrate heterogeneous components of nlp systems. In *COLING*, 1994.

[BS95] C. Boitet and M. Seligman. L'architecture en "tableau blanc": un moyen d'intégrer des composants hétérogèmes dans des systèmes de taln. In Henri Méloni, editor, *Fondements et Perspectives en Traitement Automatique de la Parole*, École Thématique, pages 257–270, 1995.

[CF95] Jean-Luc Cochard and Philippe Froidevaux. Environnement multi-agents de reconnaissance automatique de la parole en continu. In *Actes des 3èmes Journées Francophones sur l'Intelligence Artificielle Distribuée et les Systèmes Multi-agents*, pages 101–110, mars 1995.

[CLC94] Jean-Luc Cochard, Philippe Langlais, and Gérard Chollet. $\mathsf{ETC}_{\mathsf{vérif}}$, a prototype of a cooperative automatic speech recognition system. In *127th Meeting of ASA*, Boston, June 1994.

[CO95] Jean-Luc Cochard and Olivier Oppizzi. Reliability in a multi-agent spoken language recognition system. In *4th European Conference on Speech Communication and Technology*, Madrid, Spain, Sep 1995.

[CV95] Jean-Luc Cochard and Murielle Vial. Etc_vérif, a prototype of a cooperative automatic speech recognition system. In IDIAP Jean-Luc Cochard, editor, *Proc. of WOz'95: International Workshop on Oz Programming*, pages 25–33. IDIAP, Uni. Fribourg, Nov 1995.

[HSW92] Martin Henz, Gert Smolka, and Jörg Würtz. Oz—a programming language for multi-agent systems. DFKI, internal paper, 1992.

[HSW93] Martin Henz, Gert Smolka, and Jörg Würtz. Oz—a programming language for multi-agent systems. In Bajcsy [Baj93], pages 404–409.

[Lan94] Susan E. Lander. Distributed search and conflict management among reusable agents. Phd thesis, Dept of Computer Science, University of Massachusetts Amherst, May 1994.

[Smo86] P. Smolensky. Harmony theory. In The MIT Press, editor, *Parallel Distributed Processing: Explorations int eh Microstructure of Cognition*, volume 1, chapter 6, pages 194–281. D. E. Rumelhart and J. L. McClelland, 1986.

[Via95] Murielle Vial. Définition et évaluation d'un protocole de négociation dans un système multi-agents de reconnaissance de la parole. Technical report, IDIAP, Martigny, Switzerland, June, 1995.