Sonderforschungsbereich 314
Künstliche Intelligenz - Wissensbasierte Systeme

KI-Labor am Lehrstuhl für Informatik IV

Leitung: Prof. Dr. W. Wahlster

Universität des Saarlandes
FB 14 Informatik IV
Postfach 151150
D-66041 Saarbrücken
Fed. Rep. of Germany

Tel. 0681 / 302-2363

PRACMA

Bericht Nr. 109

Supporting Flexibility and Transmutability: Multi-Agent Processing and Role-Switching in a Pragmatically Oriented Dialog System

Alassane Ndiaye, Anthony Jameson

August 1994

ISSN 0944-7814

109

In Philippe Jorrand and Vassil Sgurev (Eds.), *Proceedings of the Sixth International Conference on Artificial Intelligence: Methodology, Systems, Applications*, pp. 381–390. Singapore: World Scientific Publishing, 1994.

SUPPORTING FLEXIBILITY AND TRANSMUTABILITY: MULTI-AGENT PROCESSING AND ROLE-SWITCHING IN A PRAGMATICALLY ORIENTED DIALOG SYSTEM*

ALASSANE NDIAYE ANTHONY JAMESON
Department of Computer Science, University of Saarbrücken
P.O. Box 151150, D-66041 Saarbrücken, Federal Republic of Germany
{ndiaye, jameson}@cs.uni-sb.de

ABSTRACT

Ways of achieving two desirable characteristics of pragmatically oriented dialog processing are discussed: (1) Flexible cooperation among the system's modules maximizes the system's exploitation of its knowledge and of its reasoning capabilities. (2) The ability to take either (or any) of the dialog roles in its domain enhances the system's ability to anticipate and interpret its dialog partner's reasoning and behavior. Ways of attaining these goals are being explored in the system PRACMA, which models noncooperative dialogs between a buyer and a seller. Attainment of the first goal is supported by the multi-agent architecture CHANNELS, which has been designed specifically for natural language systems. Two attempts to achieve the second goal are discussed which have been realized in two different modules of PRACMA: bidirectional, role-independent dialog planning operators; and Bayesian meta-networks for reasoning about the dialog partner's beliefs and evaluations.

1 Introduction

1.1 Issues

One key issue in developing natural language (NL) processing systems is how to find a suitable architecture that allows flexible interaction of the various modules within the system. For the NL system PRACMA [3, 10], we have integrated principles from cooperative distributed problem solving, multi-agent systems [4], and the object-oriented paradigm, to create a flexible architecture.

A second issue concerns suitable ways to make a pragmatically-oriented NL system *transmutable*, i.e. to enable it to take either of the two possible roles in a dialog. Transmutability has several advantages, one being that the system can better take into account the reasoning and the behavior of the human partner in the role it is not playing at the moment.

Possible answers concerning both issues have been realized within the NL system PRACMA. We first introduce this system and then present the CHANNELS architecture in Section 2 and our approaches to transmutability in Section 3.

*This research is being supported by the German Science Foundation in its Special Collaborative Research Program on Artificial Intelligence and Knowledge Based Systems SFB 314, Project N1, PRACMA.

1.2 The PRACMA System

Most natural language systems are restricted to cooperative dialogs. With the NL system PRACMA (**PR**ocessing **A**rguments between **C**ontroversially-**M**inded **A**ctors), we look deeper into the area of noncooperative dialog, which is widely represented in everyday situations. PRACMA models dialogs between a used-car owner and a potential buyer at a used-car market. The system has been designed to be transmutable, i.e. it is able to play the role of either the seller or the buyer.

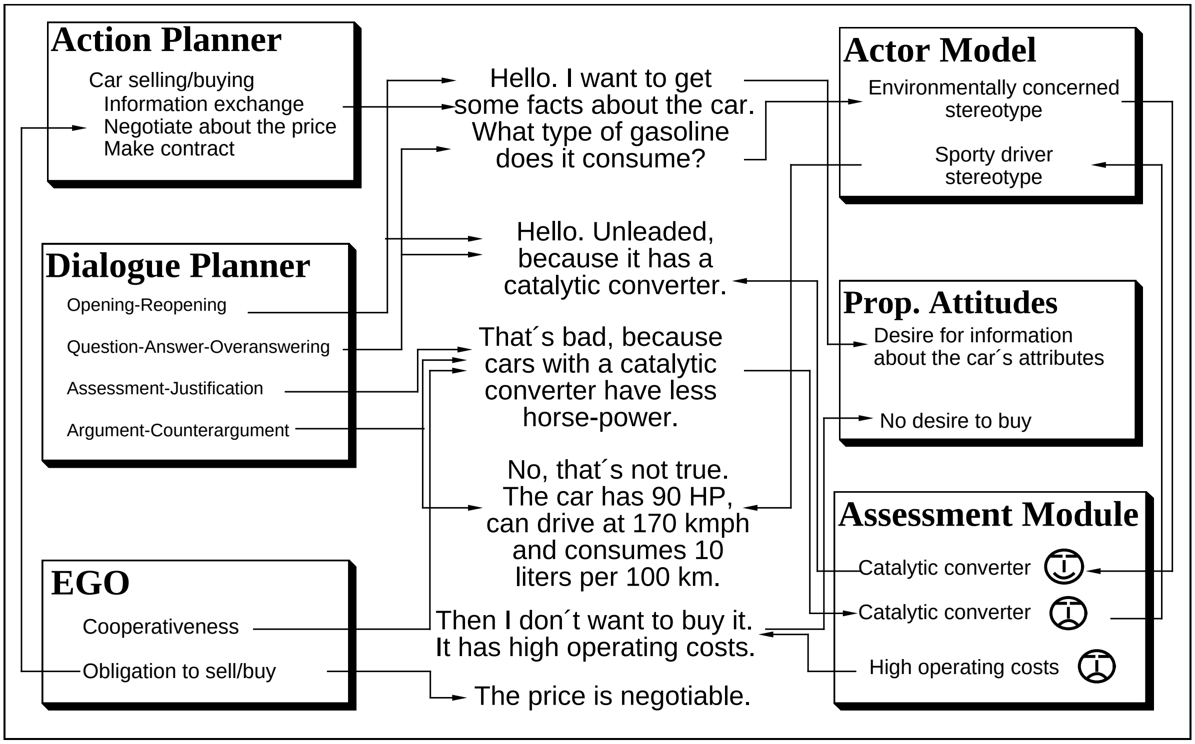

Figure 1: Schematic representation of PRACMA's processing

Figure 1 shows in the middle column an example dialog—originally in German—, surrounded by some of the processing components.[1] The arrows reflect their influence on the analysis and generation of specific dialog contributions.

Because of its primary concern with the pragmatic aspects of dialog (as opposed to syntactic and semantic aspects), PRACMA requires an architecture that allows each module to play various roles in the processing, at various stages of the processing. For example, an assessment of the buyer's interests can be used not only to determine an appropriate response to a buyer's question but also to determine the meaning of the question in the first place. The requirement of transmutability further increases the need for flexibility, as will be discussed below.

[1]Only the modules specifically needed to support the pragmatic processing are depicted.

2 Supporting Flexibility: The CHANNELS Architecture

2.1 Limitations of Some Existing NL Architectures with Respect to Pragmatic Processing

The design of the architecture of an AI system requires the decomposition of the system into modules; the specification of communication channels, communication protocols, and interaction languages between the modules; the specification of the data and control flow; and the design of the task and resource allocation strategies and the synchronization strategies.[2]

Several architectures have been proposed to meet these requirements in the context of NL processing. But many of them do not optimally support flexible pragmatic processing. The *sequential* model, which specifies a fixed, unidirectional pre-ordered connection of the various modules (e.g., as realized in XTRA [2]), is inadequate because of the very limited possibilities for communication among the modules. The main disadvantage of the *hierarchic* model is the bottleneck caused by the central control module, which handles all communication between the modules. *Cascaded* (or *pipe-line*) architectures are often used in NL systems, especially for NL generation. They consist of a sequence of modules, where each module can communicate with the next as well as with the previous one. A cascaded architecture allows for incremental and parallel processing but still permits only restricted communication. In a *blackboard* architecture (e.g., HEARSAY-II [6]) the modules—the so-called *knowledge sources*—interact asynchronously via a global data structure, the blackboard. In many blackboard systems, a central control node allocates the resources and mediates between competing knowledge sources [19]. Often there is also a centralized scheduler and a blackboard monitor.

We have found many cases in which modules for pragmatic processing can best communicate directly with one another, independent of a predefined order or a central control mechanism. Accordingly, we have developed for PRACMA the multi-agent architecture CHANNELS (**C**ooperating **H**eterogeneous **A**gents for a **N**atural-**L**anguage **S**ystem).

2.2 Agents and Messages

Each PRACMA module is modeled in CHANNELS as a (semi-)autonomous specialized problem solver, called an *agent*.[3] Each agent is characterized by, among other things, its acquaintances, its state, its skills (procedures associated with the object), and its agent model, which contains knowledge about the basic capabilities of the other agents. In addition, each agent has a table where pending messages are stored until they can be processed, as well as self-presenting capabilities.[4]

There are two basic types of communication in multi-agent systems [5]: communication via a common shared data structure (e.g., a blackboard) and communication via message passing as realized in actor languages [1]. In CHANNELS, communication and

[2]The last point is optional while the others are necessary in each AI system.

[3]In this paper, *agent* refers to an active system module, while *actor* refers to a (human or simulated) participant in the PRACMA dialog situation.

[4]Each agent is able to present its current processing state, its results, and its communication links with other agents.

interaction among the agents are achieved through a *communication-act-based protocol* which governs the exchange of messages. Each message is characterized by attributes including: the sender, the recipient(s), the type of communication act, the mode of communication (synchronous or asynchronous), the actual content of the message, and optionally the history of the message and the agents to whom the answer to the message's query should be forwarded.[5]

The communication acts [18] we currently use (*inform*, *ask*, and *reply*) define the nature of the interaction among the agents. For instance, an *ask* requests the recipient(s) to send information back to the originator of the message, while an *inform* passes information from one module to another.

Messages communicate information between a sender agent and a receiver agent either *synchronously* or *asynchronously*. The communication is synchronous if the sender requires a response before continuing processing; until a response is received, it remains in the state *waiting*. With asynchronous communication, the sender can engage in further processing before receiving a response.

For each agent there is a state transition function that determines the next state depending on its previous state, on the communication act, and on the mode of the sent or received message. The agents run concurrently as simulated parallel processes and CHANNELS uses a scheduler and the history of the messages to manage synchronization.

2.3 *Interlocking of Heterogeneous Architectures*

The CHANNELS agents can vary in granularity and complexity from very simple modules to complex architectures.[6] It is therefore possible to incorporate in a single system agents with different local architectures. For instance, the analysis module in PRACMA is realized as a blackboard, while the generator we will use has a cascaded architecture. The interlocking of heterogeneous architectures enables the reuse of previously developed modules [20]. The agents need only to be enhanced by a layer supporting the communication and cooperation with the other agents within the overall system.

2.4 *Related Work*

As mentioned above, CHANNELS integrates principles from object-oriented concurrent languages like ABCL [21]. ABCL includes objects which are viewed as autonomous information processing agents interacting with other objects solely via message passing. There are three types of message passing: *past*, *now*, *future*; and three object modes: *active*, *waiting*, and *dormant*. The *past* type (send and don't wait) corresponds to the *inform* or *asynchronous ask* in CHANNELS, while the *now* type (send and wait) is analogous to our *synchronous ask*, and the *future* type (send, specify the return value and don't wait) is similar to our *asynchronous ask*, which is always followed by a *reply*. The utility of distributed NL processing based on cooperating agents is also demon-

[5]This last concept is analogous to the concept of *reply-to* continuation in object-oriented concurrent languages.

[6]"Flexible implementation support for DAI systems must provide ways of integrating heterogeneous problem-solvers of different granularity." [8, p. 94]

strated by CARAMEL [15, 16] and TALISMAN [17]. CARAMEL is an NL system with a multi-experts architecture. For task management and control it uses several blackboards, an agenda, and a supervisor. Recently, principles of actor systems have been introduced to provide additional flexibility [16]. TALISMAN is a multi-agent system for NL processing governed by linguistic laws. It manages communication between agents without appealing to a central control mechanism. The agents communicate and cooperate only via message passing, as in CHANNELS. It is not clear, however, how the approach based on linguistic laws might be generalized to pragmatic processing. In [7], principles of a multi-agent architecture for NL processing are presented. The proposed framework features large-gained heterogeneous agents (*specialists*) which cooperate to solve an NL processing task; centralized control; and a combination of event-driven (bottom-up) and goal-driven (top-down) operation. In contrast to CHANNELS, all communication between the agents is mediated by a central *cooperation manager*; in this respect the architecture is similar to those discussed in Section 2.1.

3 Realizing Transmutability

The goal of enabling a system to take both roles within a dialog raises several issues: What are the benefits and costs of doing so? How can this be achieved efficiently and elegantly? These issues are discussed here with respect to two agents within PRACMA.

3.1 Dialog Planning Using Bidirectional Operators

PRACMA can model the dialog in Figure 1 taking the role of either the seller or the buyer. In both cases the same agents come into play, but each agent takes into account the role the system is currently taking. We discuss this transmutability using the DIALOG PLANNER agent as an example. This agent uses a hierarchical, incremental planner [13].[7] The plan operators have mostly been formulated *bidirectionally*. For instance, Figure 2 shows a part of a high-level plan operator.[8] It specifies three subgoals: to initialize the dialog, to negotiate about various aspects of the car, and to finish the dialog. The variables `?person1` and `?person2` hold information about the role (buyer or seller) that each dialog participant plays.

There are several advantages to using bidirectional dialog operators instead of two separate sets of operators, one for each role. First, the representation of the system's dialog planning knowledge is less redundant and more consistent with respect to the ways in which the two roles are handled. Moreover, it is ensured that any dialog strategy used by the system in one role can be coped with when the system takes the other role. But there are also disadvantages: A single bidirectional operator can be more complex and cumbersome than two corresponding unidirectional ones. Also, additional measures are required to do justice fully to the fact that the goals of the dialog partners conflict in part (e.g., that the invocation of a given operator may be desirable for one partner but undesirable for the other one). PRACMA's planning

[7] Actually, there are two planning processes within PRACMA that interact when performing a dialog. A comparable approach is also proposed in [11].

[8] Each expression beginning with "(AM" represents an *ask*-message that the DIALOG PLANNER sends to the **A**ctor **M**odel, an agent which models the desires and beliefs of the actors in the language MOTEL [9].

mechanisms are now being extended so as both to facilitate the writing of bidirectional operators and to increase the sophistication with which the DIALOG PLANNER uses them.

```
(define-plan-operator
  :NAME negotiation-dialog
  :GOAL
    (AM ((sb_facts am (:list (b believe all))
           (isa ?d dialog)(irole actor ?d ?person1)(irole counteractor ?d ?person2))))
  :PRECONDITIONS
    ((AM ((assert_ind (:list (b believe all)) ?d dialog)
          (assert_ind (:list (b believe all)) ?d ?person1 actor)
          (assert_ind (:list (b believe all)) ?d ?person2 counteractor)
          (sb_facts am (:list (b (believe  ?person1)(b want all))
            (isa ?d dialog)(irole actor ?d ?person1)(irole counteractor ?d ?person2))))))
  :SUBGOALS
    ((AM ((sb_facts am (:list (b believe all))
            (isa ?init initialize)(irole actor ?init ?person1)
            (irole counteractor ?init ?person2)) *optional*))
     (AM ((sb_facts am (:list (b believe all))
            (isa ?n negotiate)(irole actor ?n ?person1)
            (irole counteractor ?n ?person2)(irole topic ?n ?negoitem))) *optional*)
     (AM ((sb_facts am (:list (b believe all))(isa ?f finish)
            (irole actor ?f ?person1)(irole counteractor ?f ?person2)))*optional*)))
  ...
```

Figure 2: A part of a top-level `NEGOTIATION` plan operator

3.2 Dialog Partner Modeling with Bayesian Meta-Networks

One advantage of transmutability is due to the fact that a dialog partner often tries to anticipate and reconstruct the reasoning performed by the other partner. A system should be better able to do this if it has the necessary knowledge and dialog strategies for taking the role of its partner.

In fact, one might think that, in the role of the seller $\mathcal{S}$ for example, PRACMA could simply invoke an instantiation of itself in the role of the buyer $\mathcal{B}$ as a subroutine, so as to model $\mathcal{B}$'s reasoning. Whether this simple conception is viable we will discuss here using an example from one part of PRACMA, the agent AXIOLOGIS, which reasons about the evaluation of various aspects of the car under discussion.

3.2.1 A Simple Network for $\mathcal{B}$

The basic inference mechanism of AXIOLOGIS is the Bayesian belief network (see, e.g., [14]). The lower left-hand side of Figure 3 shows a small part of the network constructed by AXIOLOGIS when PRACMA is simulating the buyer $\mathcal{B}$ in a simple mode in which $\mathcal{B}$ takes all comments by $\mathcal{S}$ at face value. The node labeled HORSEPOWER represents $\mathcal{B}$'s *impression* about one aspect of the car which can play an important part in $\mathcal{B}$'s evaluation of the car, as illustrated in Figure 1, namely the horsepower (HP) of its motor. The first (left-hand) histogram shows that, although $\mathcal{B}$ does not know the HP precisely, he[9] has on the basis of previous experience with cars the indefinite

[9]We arbitrarily use masculine pronouns to refer to $\mathcal{B}$ and feminine pronouns for $\mathcal{S}$.

expectation that the HP is around 150.[10] The node EVALUATION OF HP represents $\mathcal{B}$'s impression of how he would evaluate the HP if he knew its exact value. The initial impression for this node (depicted by the first histogram) arises through downward propagation as soon as the initial impression for HORSEPOWER has been formed.

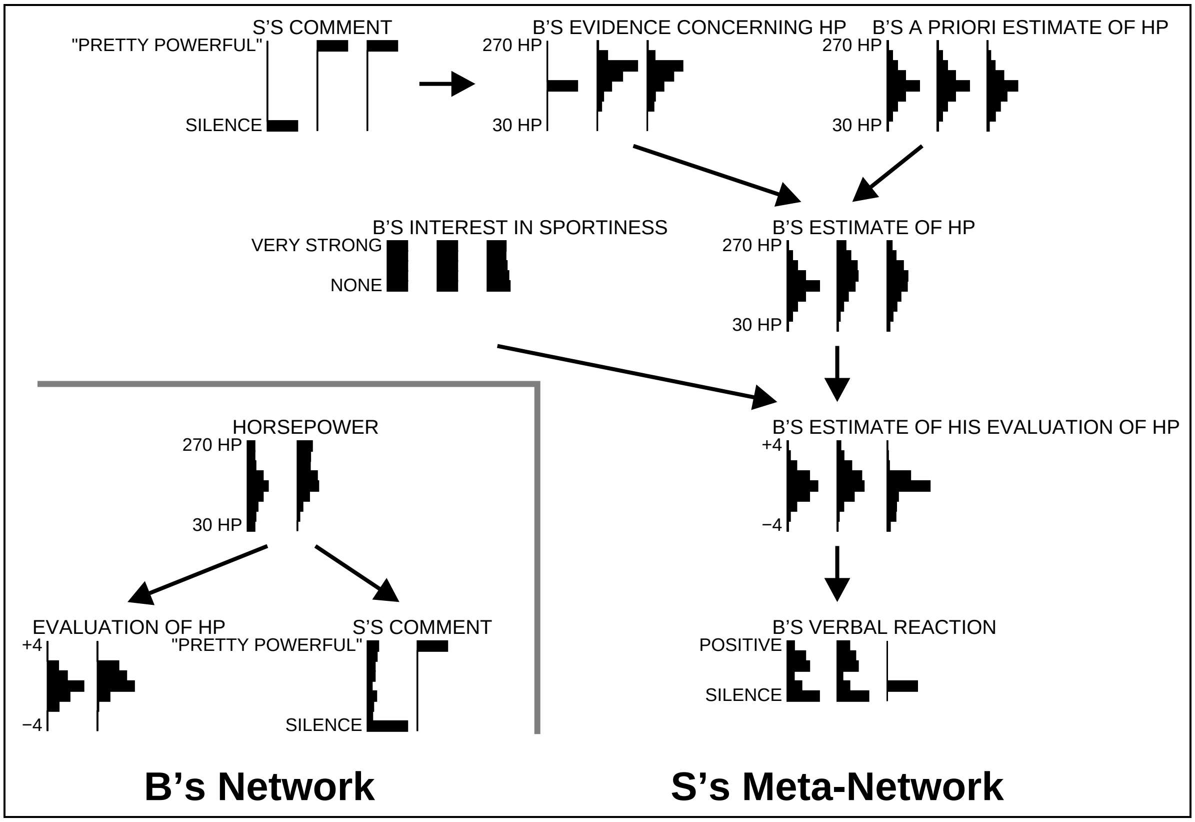

Figure 3: Illustration of the concept of a *Bayesian meta-network*

$\mathcal{B}$'s impression of the HP can be influenced by any comment $\mathcal{S}$ chooses to make about it. This dependency is represented by a parent-child link from HORSEPOWER to S'S COMMENT, because from the point of view of $\mathcal{B}$, $\mathcal{S}$'s comment is probabilistically "caused" by the truth about HP. When $\mathcal{S}$ makes a comment about HP, $\mathcal{B}$ begins its updating of this part of its network using *upward propagation* according to the usual Bayesian principles, considering those HP values more probable that would have been most likely to cause the comment to be made.[11] The change in $\mathcal{B}$'s impression of the HP is then propagated downward to the variable EVALUATION OF HP. (These changes are shown in the second histogram for each node.)

When PRACMA is taking the role of $\mathcal{S}$, it must reason about these types of probabilistic inference by $\mathcal{B}$ in order to anticipate $\mathcal{B}$'s reactions to what $\mathcal{S}$ says and to interpret $\mathcal{B}$'s behavior as reflecting $\mathcal{B}$'s underlying beliefs and interests.

[10]This variable has nine possible values, each of which represents an interval such as "255–285 HP".

[11]The relationship between the meaning of a (vague) comment and the impression change it brings about in the listener is discussed in detail in [10].

3.2.2 Approach 1: Repeated Simulations

Since PRACMA can always construct a network for $\mathcal{B}$, even when it is taking the role of $\mathcal{S}$, it could use the network directly in order to simulate $\mathcal{B}$. But even when performing an apparently simple subtask like anticipating how a given comment would affect $\mathcal{B}$'s evaluation, $\mathcal{S}$ would have to perform a number of simulations. After all, $\mathcal{S}$ has no precise information about most of the details of the network of a *specific* $\mathcal{B}$, e.g., what $\mathcal{B}$'s prior impression of the car's HP is, or how desirable $\mathcal{B}$ considers various levels of HP to be. So $\mathcal{S}$ would in general have to consider a rather large number of combinations of hypotheses about the information in the nodes of $\mathcal{B}$'s network and about the conditional probabilities that link the nodes, running a simulation for each combination of hypotheses and making inferences on the basis of the collected results. Although this approach has an attractive degree of generality and may be feasible (especially if $\mathcal{S}$ can learn from its simulations and so minimize repetitions of them), there are advantages to an alternative approach which makes more explicit $\mathcal{S}$'s uncertainty about $\mathcal{B}$'s network, namely the use of *meta-networks*.

3.2.3 Approach 2: Meta-Networks

The right-hand side of Figure 3 shows the part of $\mathcal{S}$'s Bayesian meta-network that represents $\mathcal{S}$'s (changing) beliefs about the state of $\mathcal{B}$'s network. The three nodes in the top row will be discussed below. In the second row, the variable B'S ESTIMATE OF HP represents $\mathcal{S}$'s impression of $\mathcal{B}$'s impression of HP. As it would be impractical for $\mathcal{S}$ to take into account all of the various forms of impression that $\mathcal{B}$ might have, $\mathcal{S}$ simply reasons about the single *estimate* that $\mathcal{B}$ would make if asked to guess the car's HP. The first histogram for this node reflects the impression that $\mathcal{B}$ is most likely to expect a moderately powerful motor but that $\mathcal{B}$ might also have a considerably more optimistic or pessimistic expectation. Similarly, the node B'S ESTIMATE OF HIS EVALUATION OF HP represents $\mathcal{S}$'s impression of the best guess $\mathcal{B}$ would make if asked for a single evaluation of the car's HP.

The node B'S INTEREST IN SPORTINESS is a node in the meta-network that doesn't correspond to a node in $\mathcal{B}$'s network but rather represents $\mathcal{S}$'s uncertainty about $\mathcal{B}$'s evaluation criteria for different levels of HP. (In $\mathcal{B}$'s network, these criteria are reflected in the conditional probabilities linking the nodes HORSEPOWER and EVALUATION OF HP). Specifically, the meta-network is based on the assumption that $\mathcal{B}$'s evaluation of a given HP level is a multiplicative function of his interest in the general evaluation dimension of *sportiness*.

Given prior impressions of the variables in the nodes B'S INTEREST IN SPORTINESS and B'S ESTIMATE OF HP, $\mathcal{S}$ forms an impression of B'S ESTIMATE OF HIS EVALUATION OF HP using the same general propagation principles as those used in $\mathcal{B}$'s network—though of course the conditional probabilities that link the nodes in this meta-network are different from the ones linking the corresponding nodes in $\mathcal{B}$'s network. As the node B'S VERBAL REACTION shows, $\mathcal{S}$ can also go on to predict what type of utterance (if any) $\mathcal{B}$ might make reflecting his evaluation of the car's HP[12].

[12]The conditional probabilities used here were derived from an unpublished empirical study involving the role-playing of sales dialogs.

The top row of the meta-network shows that actually three nodes in all are needed to represent $\mathcal{S}$'s impressions of $\mathcal{B}$'s impression of the car's horsepower, if $\mathcal{S}$ wants to be able to *manipulate* $\mathcal{B}$'s impression by supplying $\mathcal{B}$ with relevant evidence. The three nodes B'S EVIDENCE CONCERNING HP, B'S A PRIORI ESTIMATE OF HP, and B'S ESTIMATE OF HP reflect the general principle that, for each node in a Bayesian network, to allow both upward and downward propagation it is necessary to store not just a single probability distribution representing a belief about the variable but also two independent distributions summarizing both the prior expectation for the variable and the relevant evidence due to observations[13]. So the top row of the meta-network reflects the fact that $\mathcal{S}$ cannot influence $\mathcal{B}$'s *a priori* impression about the car's HP but can provide $\mathcal{B}$ with specific *evidence* concerning this variable and thereby (within certain limits) influence $\mathcal{B}$'s overall impression of the HP. (The second histogram for each node shows $\mathcal{S}$'s impression for that node after $\mathcal{S}$ has made a positive, though vague comment: "The motor is pretty powerful".)

The example just mentioned involves *downward* propagation in $\mathcal{S}$'s meta-network for the purpose of anticipating the *upward* propagation that will occur within $\mathcal{B}$'s network. But of course $\mathcal{S}$'s own network can itself exhibit upward propagation. The third histogram for each node shows how $\mathcal{S}$ adapts her impressions of $\mathcal{B}$ if he produces an unexpectedly negative utterance about the car's HP (e.g., "The horsepower is a problem"). The only really obvious change is that B'S ESTIMATE OF HIS EVALUATION OF HP now seems very unlikely to be positive. Close inspection of the histograms higher up in the meta-network reveals that $\mathcal{S}$ distributes the "blame" for $\mathcal{B}$'s negative evaluation over several sources: $\mathcal{B}$ now appears slightly less interested in sportiness (B'S INTEREST IN SPORTINESS), his prior impression of the motor's power was perhaps more pessimistic than $\mathcal{S}$ originally suspected (B'S A PRIORI ESTIMATE OF HP), and he may have interpreted $\mathcal{S}$'s vague comment about the motor being "pretty powerful" relatively conservatively (B'S EVIDENCE CONCERNING HP).

In sum, meta-networks allow PRACMA to make some inferences about the dialog partner's processing which are intuitively familiar from everyday experience (though some have rarely, if ever, been handled by previous dialog systems), and to do this using a relatively explicit, straightforward representation[14]. But the approach raises a number of rather complex issues, and further investigation of these may lead to various changes in the form of PRACMA's meta-networks. For example, there are alternative ways of conceptualizing a meta-node such as B'S ESTIMATE OF HP; and it may prove unnecessary to represent the meaning of a comment explicitly with a meta-level node like B'S EVIDENCE CONCERNING HP.

References

[1] G. Agha. *The Structure and Semantics of Actor Languages*. In J. W. de Bakker, W. P. de Roever, and G. Rozenberg, (eds.), Foundations of Object-Oriented Languages, pp. 1–59. Springer, Berlin, 1991.

[2] J. Allgayer, R. Jansen-Winkeln, C. Reddig, and N. Reithinger. *Bidirectional Use of Knowledge in the*

[13] This information is in fact stored for each node in $\mathcal{B}$'s network, though only the overall probability distribution, which is derived from the two others, is depicted.

[14] In fact, $\mathcal{S}$'s meta-network in turn serves as a basis for a meta-meta-network that models the inferences of a sophisticated $\mathcal{B}$; discussion of this would exceed the scope of the present paper.

Multi-Modal NL Access System XTRA. In Proc. of the Eleventh IJCAI, pp. 1492–1497, Detroit, MI, 1989.

[3] J. Allgayer, A. Kobsa, C. Reddig, and N. Reithinger. *PRACMA: Processing Arguments Between Controversially-Minded Agents*. In Proc. of the Fifth Rocky Mountain Conference on Artificial Intelligence: Pragmatics in Artificial Intelligence, pp. 63–68, Las Cruces, NM, 1990.

[4] A. Bond and L. Gasser. *Readings in Distributed Artificial Intelligence*. Morgan Kaufmann, San Mateo, CA. 1988

[5] A. Cawsey, J. R. Galliers, S. Reece, and K. Sparck Jones. *A Comparison of Architectures for Autonomous Multi-Agent Communication*. In Proc. of the Tenth ECAI, pp. 249–251, Vienna, 1992.

[6] L. D. Erman, F. Hayes-Roth, V. R. Lesser, and D. R. Reddy. *The HEARSAY-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty*. In B. L. Webber and N. J. Nilsson, (eds.), Readings in Artificial Intelligence, pp. 349–389. Morgan Kaufmann, Los Altos, CA, 1981.

[7] D. Fum, G. Guida, and C. Tasso. *A Distributed Multi-Agent Architecture for Natural Language Processing*. In Proc. of the Twelfth COLING, pp. 812–814, Budapest, 1988.

[8] L. Gasser and J.-P. Briot. *Object-Based Concurrent Programming and Distributed Artificial Intelligence*. In N. M. Avouris and L. Gasser, (eds.), Distributed Artificial Intelligence: Theory and Praxis, pp. 81–107. Kluwer, Dordrecht, 1992.

[9] U. Hustadt and A. Nonnengart. *Modalities in Knowledge Representation*. In Proc. of the Sixth Australian Joint Conference on Artificial Intelligence, pp. 249–254, Sydney, 1993.

[10] B. Kipper and A. Jameson. *Semantics and Pragmatics of Vague Probability Expressions*. In Proc. of the 16th Annual Conference of the Cognitive Science Society, Atlanta, GA, 1994.

[11] L. Lambert and S. Carberry. *Using Linguistic, World, and Contextual Knowledge in a Plan Recognition Model of Dialogue*. In Proc. of the 14th COLING, pp. 310–316, Nantes, 1992.

[12] J. Laubsch and J. Nerbonne. *An Overview of $\mathcal{NLL}$*. Technical Report, HP Labs, 1991.

[13] J. D. Moore and C. L. Paris. *Planning Text for Advisory Dialogues*. In Proc. of the 27th Annual Meeting of the ACL, pp. 203–211, Vancouver, 1989.

[14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1991. (Revised second printing).

[15] G. Sabah. *CARAMEL: A Computational Model of Natural Language Understanding Using a Parallel Implementation*. In Proc. of the Ninth ECAI, pp. 563–565, Stockholm, 1990.

[16] G. Sabah and X. Briffault. *CARAMEL: A Step Towards Reflection in Natural Language Understanding Systems*. In Proc. of the Fifth IEEE International Conference on Tools with AI, Boston, 1993.

[17] M.-H. Stefanini, A. Berrendonner, G. Lallich, and F. Oquendo. *TALISMAN: Un système multi-agents gouverné par des lois linguistiques pour le traitement de la langue naturelle*. In Proc. of the 14th COLING, Nantes, 1992.

[18] D. Vanderveken. *Meaning and Speech Acts*. Cambridge University Press, Cambridge, 1990.

[19] H. Velthuijsen. *The Nature and Applicability of the Blackboard Architecture*. Doctoral Dissertation, University of Maastricht, 1992.

[20] T. Wittig, (ed.). *ARCHON: An Architecture for Multi-Agent Systems*. Ellis Horwood, London, 1992.

[21] A. Yonezawa (ed.). *ABCL – An Object-Oriented Concurrent System*, The MIT Press, Cambridge, MA, 1990.