A Generic Model for Intelligent Negotiating Agents

Brigitte Lâasri*, Hassan Lâasri*, Susan Lander, and Victor Lesser
Computer and Information Science Department
University of Massachusetts
Amherst, Massachusetts, USA 01003

Abstract

Research in Cooperative Distributed Problem Solving (CDPS) considers how problem-solving tasks should be allocated among a group of agents and how the agents should coordinate their actions to achieve effective problem solving. For some CDPS systems, negotiation plays an important role in how agents cooperate. We define negotiation as the process of information exchange by which the agents act to resolve inconsistent views and to reach agreement on how they should work together in order to cooperate effectively. We describe a generic model, the *Recursive Negotiation Model* (RNM), that can serve as a basis for classifying and specifying where conflict resolution among multiple experts, viewpoints, or types of reasoning is needed in building a sophisticated CDPS system. This model defines where and how negotiation can be applied during problem solving based on structuring problem solving into four stages: *problem formulation*, *focus-of-attention*, *allocation* of goals or tasks to agents, and *achievement* of goals or tasks. We further discuss how the degree of agent participation in control decisions, including decisions about assigning responsibility to agents, influences the nature of negotiation within a particular system. Through this model, we emphasize that negotiation may be a recursive, complex, and pervasive process that is used to resolve conflicts in both domain-level and control-level problem solving. Finally, we survey existing negotiation frameworks and how they relate to our generic model.

*Current Address: Alcatel Alsthom Recherche, Route de Nozay, 91460 Marcoussis, France.

Contents

1 Introduction

Cooperative Distributed Problem Solving (CDPS) is concerned with how a set of agents can work together to solve problems that are beyond their individual capabilities [10]. In this paper, we assume that each agent is capable of sophisticated problem solving and can work semi-autonomously. We also assume agents work *cooperatively* by (1) using their local expertise, resources and information to formulate partial solutions and by (2) integrating their solutions with those of other agents in order to build an overall solution that meets the goals of the system. In conflict situations, the achievement of an acceptable overall solution may involve agents giving up or only partially satisfying some goals. These goals can be local to the individual agent(s) or non-local, spanning multiple agents.

Some problems may involve agents with different perspectives working on the same goal or agents working on interdependent goals which share constraints or require common and limited resources for their resolution, i.e., explicit or implicit interacting subproblems. Thus, agents may face conflicting solutions to goals during problem solving; additionally, some of these conflicting solutions may relate to meta-goals (control decisions) about the most effective problem-solving organization, the set of goals to solve, the assignment of goals to agents, and the order in which to solve the goals. Conflicts may arise even if the agents try to coordinate their problem solving due to their lack of an updated and complete view of the problem-solving state of other relevant agents, due to the distribution of information and limited communication bandwidth, or due to their heterogeneous expertise or evaluation criteria. As in a human organization, agents can deal with such conflicts through *negotiation*, the process by which the agents act to resolve inconsistent views and to reach agreement on how they should work together in order to cooperate effectively.

This paper analyzes negotiation both from conceptual and computational viewpoints[1]. We describe a generic model, the *Recursive Negotiation Model* (RNM), that can serve as a basis for classifying and specifying where conflict resolution among multiple experts, viewpoints, or types of reasoning is needed in building a sophisticated CDPS system. It also categorizes the various styles of negotiation that could occur depending on (1) whether or not negotiation is an integral part of the problem solving, (2) the particular problem-solving stage where negotiation may be used, and (3) the assumptions on which the agents' cooperation is based. Through this work, we emphasize that negotiation may be a recursive, complex, and pervasive process that is relevant not only to domain problem solving but also to control problem solving.

In Section 2 we describe, from a goal-oriented perspective, the problem-solving stages that multi-agent systems must pass through to cooperatively solve a complex problem. Section 3 discusses how the various stages of problem solving are controlled from an agent-oriented perspective. In Section 4, the role of negotiation in CDPS systems is examined based on the framework for viewing cooperative problem solving developed in the previous sections. Mechanisms for information sharing and conflict resolution are discussed. Section 5 describes a set of systems that are representative of current approaches to negotiation in CDPS systems and how their use of negotiation fits into our general model.

2 CDPS from a Goal-Oriented Perspective

We have identified four stages that occur in multi-agent problem solving. For a given system, some of these stages may be implicitly embedded in the architecture or inference engine of the system while some will be explicitly acknowledged as a separate stage. We introduce these stages through a simple example from engineering design.

Suppose that a set of agents are designing a complex artifact. At the beginning of problem solving, the agents have to identify the components of the artifact. In some systems, the system designer provides this information. Designing a complete system is the *top-level* goal[2] that must be achieved in order to solve the problem. This goal can be decomposed into a set of subgoals, each to design one component of the system. Dependency relationships among the goals are also specified in the decomposition. In this case, a conjunctive relationship exists between the subgoals for the design of each component (all components must be designed in order for the complete system to be designed). Additional dependency relations among goals arise when design components have shared parameters whose solution values must be consistent; these shared parameter values define the interface connections among components. This stage, where the current set of goals is formulated, is called *problem formulation*.

There may be insufficient resources to design all the components simultaneously and/or there may be enabling constraints among component designs. Therefore, there is a need for a *focus-of-attention* stage; each time the agents pass through it, they select the set of components to design next. Each component in this set represents an *active* goal that the agents will work on next.

Deciding which agents will design which components of the selected set constitutes the *allocation* stage of active goals to agents. The agent[3] responsible for a selected component may gener-

[1] However, we don't take into account human factors described in [25].

[2] Many multi-agent systems are task-oriented rather than goal-oriented. When the term *goal* is used with respect to the general model, it is often more appropriate to think in terms of *tasks* for a specific system.

[3] We use the term agent for convenience but recognize that more than one agent may be responsible for a selected component.

ate a design directly or may further decompose the component. In the latter case, the goal is decomposed into a set of either independent or interacting subgoals. This allocation/decomposition[4] cycle is obviously a recursive process, but one that will eventually result in the assignment of a set of directly achievable goals to a set of agents that are capable of achieving them.

In the *achievement* stage, an agent directly attempts to achieve an allocated goal. If the agent succeeds in designing the component, the goal is achieved; otherwise, the agent faces a failure. To overcome this situation, the agent (possibly in consultation with other agents) can relax local constraints, change local evaluation criteria, or try an alternative problem-solving method. If this isn't possible, the goal cannot be satisfied in its present form and control returns to a higher level in the problem-solving hierarchy, possibly with information about the cause of the failure. At the higher level, it may be possible to reorganize the supergoal (find a new decomposition, for example) or to reallocate the failed goal to another agent.

Since multiple agents may design subcomponents of the same component, there may be multiple ways of integrating their results. For example, the synthesis of results may vary with the order in which the results are aggregated, with the properties of these results, and with the agents that generated them. Just as there may be a need to allocate domain-level subgoals, there may be a need to allocate goals specifically to synthesize the results.

Negotiation may occur among agents at any of these stages. *Peer* negotiation takes place when multiple agents resolve conflicts at the same level in the problem structure. *Hierarchical* negotiation occurs when agents at different levels in the problem structure must resolve conflicts. An example of peer negotiation is when two agents are designing components at the same level and must resolve a conflict over the value of some shared parameter. An example of hierarchical negotiation is when an agent designing a subcomponent is unable to complete its task under the current specification and negotiates with the allocating agent to change the specification. Note that in this hierarchical case, the allocating agent may need to negotiate with its peers in order to make changes in the specification or it may choose to reallocate rather than negotiate.

In the rest of the paper, we will use the following terms in the discussion of the dynamic problem-solving structure. We define a *top-level* goal to be a goal shared by all agents. A *local* goal is a goal that only one agent is trying to achieve, whether it is externally assigned or whether it is part of the internal specification of the agent. A *common* goal is any goal shared by a subset of the full agent set. When we use the term "goal" without any further definition, we are referring to a common goal. Similarly, we describe evaluation criteria for solutions as either *local*, *common*, or *top-level*. Local evaluation criteria define the utility of local, possibly unshared, solutions of an agent. Common evaluation criteria define the utility of the integrated solutions of some subset of agents. Top-level evaluation criteria are those solution requirements that define the utility of a complete solution based on attribute values of that solution. Those values may be synthesized from numerous local or common solutions.

We now describe in detail the four stages that occur as: 1) goals are specified, *problem formulation*; 2) problem-solving activity is focused, *focus-of-attention*; 3) goals are allocated to agents, *allocation*; and 4) goals are achieved, *achievement* (see Figure 1).

Problem Formulation: In this stage, the goals or tasks that represent the requirements for a particular level of problem solving are identified and a problem-solving structure is built. We distinguish three types of problem formulation. The first type is a top-down *decomposition*

[4]Decomposition, as part of the allocation stage, is similar to problem formulation but takes place only in a single agent.

process where the original problem is decomposed into a set of subproblems, either to achieve the satisfaction of some goal (or accomplishment of some task) or to synthesize results from the achievement of goals (or tasks) at some lower level. The second type of problem formulation, *composition*, is a bottom-up process that creates supergoals in response to a data-driven problem-solving methodology. For example, if a set of opportunistic agents responds to a particular set of input data, the synthesis of their results may also need to be opportunistically determined, i.e., determined from the specific results that are available [8]. The third type of formulation, *reorganization*, is used to choose an alternative set of subproblems or superproblems when a failure has occurred in the current set.

Dependency relationships among goals or tasks are determined as part of the problem structure. These relationships are classified as: *or*, indicating disjunctive subgoal alternatives, *and*, indicating conjunctive subgoal achievement is required, and *overlapping*, representing the situation where both *or* and *and* relationships exist simultaneously. More complex relationships can also exist [7, 32]. For example, in temporal reasoning, it is sometimes necessary to specify relationships such as *Task A must start after Task B starts but before Task B finishes.*

Peer negotiation can occur at the problem formulation level if multiple agents are involved in the formulation process and have different perspectives on how best to accomplish it. Hierarchical negotiation can occur downward, in response to requests from agents responsible for focus-of-attention, allocation, or achievement, resulting in a reorganization of the problem structure.

Focus-of-Attention: Once a problem-solving structure has been initially formulated, a (sub)set of goals is selected to be the current focus-of-attention of a set of agents. When there are limited resources for problem solving or enabling constraints in the problem structure, there is a need to determine which goals to work on next. The selected goals are referred to as *active* goals. Peer negotiation can occur if multiple agents are selecting active goals and have conflicting viewpoints on how to accomplish the selection. Hierarchical negotiation can occur either upward, negotiating a problem reorganization, or downward, refocusing in response to a request from agents that are attempting to allocate or achieve focused goals.

Allocation: In this stage, active goals are allocated to agents to be achieved or further decomposed or composed. Appropriate agents can be determined based on characteristics of the problem and agent set including: each agent's available resources, the current load distribution, specific expertise of the agents, an agent's proximity to required data, or an organizational structure. Peer negotiation may occur if multiple agents are allocating goals. If the agents that can solve the goals have redundant capabilities and multiple agents can take responsibility for the same goals, the allocating agents may negotiate over the most effective allocation. Hierarchical negotiation can occur upward, changing the set of active goals to be allocated or reorganizing the problem-solving structure. It can also occur downward in response to a request from agents that are attempting to achieve allocated goals.

Achievement: Each allocated goal is assigned to one or more agents. This agent set may attempt to achieve the goal directly or it may further decompose or compose it. If an agent has more than one allocated goal, it may make additional focus-of-attention decisions locally to determine the order in which to process those goals. Failures can occur when some agent cannot satisfy a goal due to lack of required knowledge or data, when the problem is overconstrained, or

when multiple agents have inconsistent or incomplete knowledge, requirements, data, or evaluation criteria. Peer negotiation can be applied to failures that result from conflicts among agents. Hierarchical negotiation can occur upward, either changing the allocation of goals to agents, changing the set of focused goals, or reorganizing the problem structure.

Flow of Control Among Stages

The problem (re)formulation stage occurs when: (1) problem solving begins to generate the first level of goals; (2) agents further decompose or compose goals that have been allocated to them; or (3) agents face a failure at some other stage.

The focus-of-attention stage occurs when (1) new goals are generated; (2) active goals are achieved or deleted; or (3) active goals cannot be allocated or achieved. In the latter case, changing the order in which existing goals are processed (changing the focus-of-attention) may cause some previously unachievable goals to be solvable. It may also enable the allocation of some previously unallocated goals. Allocation occurs (1) each time new goals are activated; or (2) when goals cannot be achieved by the agents to which they were allocated.

Each time a failure occurs at any level, a decision is required about how to proceed. For example, if a particular goal cannot be achieved by the responsible agent, it might be because the agent's solution to this goal conflicts with another agent's viewpoint. By exchanging knowledge and data, it may be possible to directly resolve the conflict by peer negotiation at this level. When a conflict is resolved at this level: 1) the problem does not need to be reallocated, refocused, or reformulated, any of which can be expensive, time-consuming, and can result in previous work becoming obsolete; and 2) it may provide future benefits: the agents may be able to use the information exchanged during the resolution to guide future processing. On the other hand, it may be that solving the problem under the constraints imposed by the current problem-solving structure will result in consistently poorer solutions than could be obtained under some other structure. The decision about how much work should be done to resolve the problem at each level is difficult since the costs and benefits of reachievement, reallocation, refocusing, and reformulation must be accurately estimated.

When a conflict has occurred and been resolved by negotiation, it may mean that some agents have agreed to accept solutions that they know are not optimal by their own standards, i.e., they have agreed to relax some local solution requirement. Any time a solution requirement has been relaxed, it is believed that either the problem is overconstrained under the current problem-solving structure, or it is highly unlikely that a solution will be found and that further effort to search for one will lead to an unacceptable amount of computation. In either case, it may not be known if another structure exists that might remove the conflict. Further, it may not be known how to relax local requirements so that the best possible overall solution will be generated. In fact, there may not be any method for measuring optimality of overall solutions, or there may be some method that is separate from any local measurements of optimality. Given these situations, knowing when to accept a proposed solution and terminate the problem solving is extremely problematic. In general, it will not be possible to explore every option at every level and it cannot be known whether some other problem-solving structure, focus, allocation, or compromise between peers might result in a better solution and what the cost would be to find that solution. Thus, in a CDPS system which has this wide range of alternative decisions, the concept of satisficing as a criterion for judging solution acceptability is often used rather than optimality [10, 21, 23].

Discussion

A problem-solving stage is characterized by: (1) its type, one of the four cited above, (2) the set of agents that perform it, (3) its context: a goal or task (or a set of goals or tasks), and (4) its results. As shown in the design example, this process is recursive: subproblems resulting from a problem decomposition are solved through the same stages as the original problem. In the same manner, superproblems that synthesize results of a set of problems involve the same stages. Multiple instances of the same stage or of different stages may be present during problem solving. For example, some agents may be decomposing a goal while others are composing a supergoal or some agents may be attempting to achieve a goal while others are selecting active goals. Furthermore, an agent may have various tasks to perform at the same time, each one relating to a different stage. For example, an agent may have a goal to decompose and another one to allocate. Thus the stages are interleaved during problem solving.

There is, however, an implicit partial order in which the stages proceed with respect to a specific goal. This order is based on the relationships among the stages discussed earlier and shown in Figure 1. Consider a goal and suppose that this goal is in a state of being further decomposed into common subgoals by a set of agents. From this goal perspective, the various stages of processing the goal will be as follows: some of the generated subgoals will become active and will be allocated to agents. Unless they can be directly achieved by the assigned agent set, each active common subgoal will be further decomposed into subgoals, and the process repeats.

The engineering design example showed how RNM handles a single top-level goal (the entire problem). The goal was implicitly active and allocated to all agents. This isn't a restrictive assumption. To the contrary, when dealing with multiple top-level goals the process is as follows: selection of active top-level goals, allocation of those goals to the appropriate agents, and achievement of the goals which means formulation of each one of them. In fact, the process starts as if the global problem was originally formulated.

In [1], Bond and Gasser divide the DAI world into two primary arenas. They define *Distributed Problem Solving (DPS) systems* as those systems concerned with how the work of solving a particular problem can be divided among a number of problem solvers that cooperate at the level of dividing and sharing knowledge about the problem and about the developing solution. They defined *Multi-Agent (MA) systems* as systems concerned with coordinating intelligent behavior among a collection of possibly pre-existing autonomous intelligent agents to jointly share their goals, take action or solve problems. The first type of systems can be viewed as "problem-driven" systems in the sense that the resulting system architecture is based on the problem at hand while the second type of systems can be viewed as "agent-driven" systems in the sense that the agents are pre-existing and may have their own goals (e.g., specialized robots) and the issue is how these agents co-exist in the same environment.

RNM can be considered as a first step of unifying these two arenas: it handles DPS systems with a single global goal as well as MA systems with multiple shared goals. In a similar manner, RNM provides a unified framework for viewing CDPS that integrates the two styles of cooperation—"task-sharing" and "results-sharing"—studied by Smith and Davis [29]. "Task-sharing" deals with problem solving through decomposition where agents share the computational load for the execution of subtasks of the overall problem. In the "results-sharing" form of cooperation, agents share results and have to cooperatively build an overall solution. This cooperative building may involve a composition process in which supergoals are dynamically created.

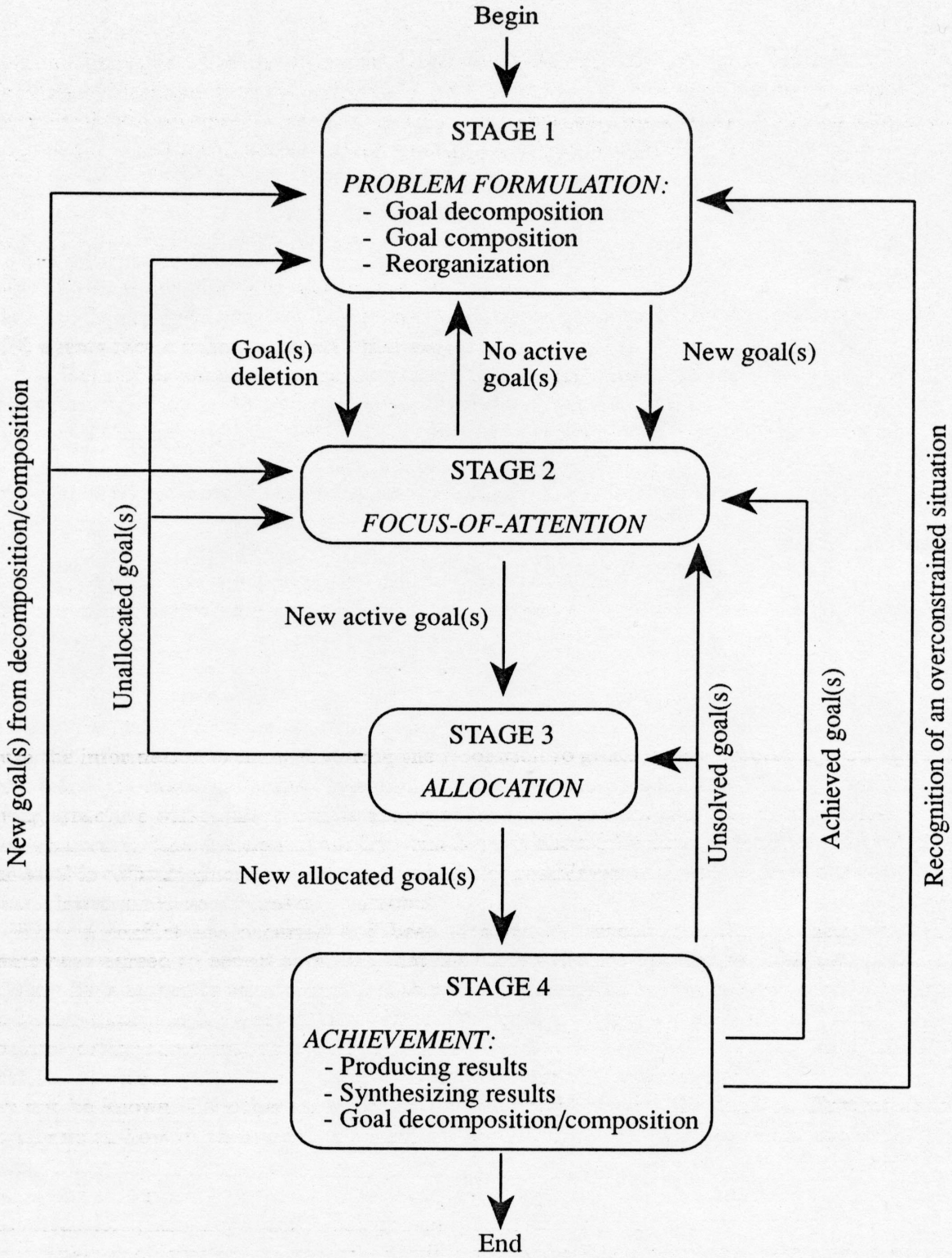

Figure 1: A goal-based view of CDPS problem-solving stages

3 CDPS from an Agent-Oriented Perspective

In the previous section, we defined the problem-solving stages through which goals are processed. We showed that there is an implicit partial order in processing goals but that this partial order is viewed only from a goal perspective. From an agent perspective, there may be multiple instances of the stages distributed among the agents and going on simultaneously (as in the contract net model in [5]); these instances come from various levels of the problem structure. Having defined these basic stages, we proceed to describe ways that each stage can be implemented within an agent set. In many systems, some of the stages will be implicitly defined, some will be explicitly defined but external to the normal course of problem solving, and some will be defined as an integral part of the problem-solving methodology. In the latter case, the problem arises as to *what stage instance an agent will focus on next.* In this way, focus-of-attention can be concurrently going on from both a goal and agent perspective.

A goal-oriented approach to dealing with this meta-level control issue is to consider the stages as *meta-level goals* that the agents can explicitly reason about. Likewise, the agents can manipulate these goals and cooperate to achieve them. In the rest of the paper, the term goal includes also meta-level goals.

Meta-level control knowledge permits the agents to examine the current problem-solving situation (goals being solved, goals to solve, time spent in the current problem solving, etc.) and decide what stage should be tackled. This is in contrast to the use of control knowledge to decide what actions should be taken (for example, selection of active goal).

An important consequence of meta-level control is the degree of sophistication of the agents: each agent must be capable of reasoning about its own problem solving, how this fits in with problem solving of other agents, and what it can do to improve group problem solving. The result of sophistication is twofold. On one hand, there is a risk that adding meta-level control introduces overhead which might not justify the resulting sophistication. On the other hand, this provides the agents with more "rationality" and control over their local problem solving as well as over their collaborative work [8]. As more fully discussed in [10, 21], sophisticated local control in conjunction with the exchange of meta-level control information is an important feature for agents involved in complex or ambiguous cooperative problem solving. It allows an agent to understand the implications of its planned problem solving and communication actions for other agents' goals, beliefs, and plans. This understanding forms the basis for deciding how to coordinate and negotiate with others.

3.1 Agent Participation in Control

Let G designate a goal and G^1 designate the set of goals resulting from the formulation stage of G. Formulation can be done by:

1. An implicit process *outside* the scope of the agents: G^1 is predefined.
2. An explicit process *outside* the scope of the agents: G^1 is determined by a framework that supports the integration of the agent set.
3. A *central* agent: G^1 is determined explicitly by a single agent.
4. A set of *distributed* agents: G^1 is determined explicitly by multiple agents.

Each of the previous techniques can be used simultaneously for generating different classes of goals. For example, top-level goals may be predefined while all other goals are formulated by distributed agents.

The discussion above centers around problem formulation, but each of the problem-solving stages can be addressed in a similar fashion. Let G^2 represent the set of goals selected during the focus-of-attention stage. G^2 can be determined by: 1) an implicit process outside the scope of the agents; 2) an explicit process outside the scope of the agents; 3) a central agent; or 4) a set of distributed agents. This also holds true for both the *allocation* and *achievement* stages. In the achievement stage, one would normally expect goal achievement to be accomplished by the agents in the agent set. However, some of the basic goals may be achieved by humans in a *computer-supported cooperative work* environment [15, 16].

3.2 Agent Participation in Assigning Responsibility

Consider again the goal formulation stage for G. Unless this stage is implemented as a process outside the scope of the agents, there is a need to determine which agent(s) will formulate the subgoal set, G^1, from G. Let A^1 represent these agents. They can be determined by:

1. An implicit process *outside* the scope of the agents: A^1 is predefined or A^1 is determined by a static organizational structure imposed on the agent set.

2. An explicit process *outside* the scope of the agents: A^1 is determined by a framework that supports the integration of the agent set.

3. A *central* agent: A^1 is determined explicitly by a single agent.

4. A set of *distributed* agents: A^1 is determined explicitly by multiple agents.

For the last two cases, a question arises: *Which agent(s) will select the set of agents,* A^1? Let $A^{1\prime}$ represent these agents. $A^{1\prime}$ might be the agents that were responsible for achieving G and decided to further decompose it. On the other hand, $A^{1\prime}$ might comprise agents that are organizationally motivated rather than motivated by direct involvement in the problem-solving process, i.e., an agent organization may exist in which a separate set of agents is responsible for choosing the agents that will manipulate the goals during each stage of processing. For example, there might be a "committee" established that uses information about the work load and performance capabilities of each agent to evaluate the situation and make assignments at the start of each stage of problem solving. A single committee might make all assignments or there might be separate committees for each stage. The committee itself might be a single or multiple agents or a process outside the scope of the agents.

The same question arises in the other stages of G's processing. If G's focus-of-attention stage is achieved by A^2, a likely candidate for the determination of A^2 is A^1. In the same manner, the most likely candidates for determining A^3, the agents responsible for allocating G's active goals, are either A^1 or A^2. Again, however, it may be that direct involvement in problem solving is not the determining factor and instead an agent organization exists that specifies these agents directly.

4 Negotiation in RNM

Negotiation occurs in situations where potential conflicts exist among agents due to their *interdependent* activities. Agents entering into a negotiation will try to avoid potential conflicts

and resolve conflicts that have occurred even at the expense of their specific and individual interests if necessary. Thus, once agents agree to negotiate, they have agreed to behave as cooperatively as possible because of their common interest in solving the conflict. Drawing from experience with human organizations and team problem solving, it is reasonable to minimize conflict within an organization of agents. This can be done through *a priori* specification of relevant team attributes: goals, duties, hierarchies, and organizational structure for example. However, for particular types of problems, applying a rigid definition of acceptable behavior at some stage of problem solving will lead to unacceptable brittleness. Therefore, negotiation is a well-suited coordination process for CDPS systems where agents act cooperatively.

We have identified two forms of negotiation. We see negotiation as an integral part of CDPS if agents cooperate through exchange and evaluation of possibly conflicting partial results generated during local problem solving[5]. On the other hand, we consider negotiation as outside normal problem solving if agents interact to resolve conflicts only after independently completing their local problem solving. In the first case, domain-level problem-solving tasks are interleaved with and appropriately structured to account for the communication and assimilation of information that can be used to detect conflicts and guide their resolution. In the second case, no attempt is made to exchange or assimilate external information until after the involved agents have generated their results.

As shown, during the formulation of goals, multiple agents may cooperate in defining G^1 to work on or in selecting agent(s) A^1. It is important to note that unless all of the agents have access to the same information and algorithms, negotiation may be needed to resolve potential conflicts among the agents (multiple agents may not have the same view of how to organize the solving of a common goal and thus will infer different or even contradictory goal hierarchy structures).

More generally, for each stage, negotiation can occur when multiple agents are solving interdependent goals and have different knowledge, reasoning mechanisms, or evaluation criteria for solutions. In these cases, agents may infer different or even contradictory decisions and thus enter into conflicts. Negotiation results in (1) the recognition of an overconstrained situation if there is no mutually acceptable answer, (2) the generation of a solution *completely* acceptable to all parties involved, or (3) the generation of *satisficing* solutions that are less than optimal but still acceptable. In case 3, there are two ways of generating such satisficing solutions: a) relaxing local solution requirements in order to reprocess earlier rejected solutions or to generate new ones; or b) reorganizing the original problem in a way that is likely to be solvable (reformulating the problem).

4.1 Exchanged Information

Negotiation is based on formulation and exchange of information especially in terms of *proposals*, *critiques*, *explanations*, and *meta-information* as described below.

4.1.1 Proposals

By a proposal, we mean a *specific solution*, a *partial result*, a *cluster solution*, or a *partial cluster*. To be more concrete let us for example represent the solution space as a three-dimensional space (x, y, z). In this example, a specific solution would be a point (x_0, y_0, z_0); a partial result would

[5]This form of negotiation is very similar to cooperative problem solving described as Functionally-Accurate, Cooperative (FA/C) [21].

be, for instance, $x = x_0$ which means that the solution is constrained to be on the plane (x_0, y, z); a cluster solution would be $x = x_0$, $a < y < b$, and $c < z < d$ which provides a range of acceptable solutions; finally, a partial cluster would be $x = x_0$ and $a < y < b$ which means a range of acceptable partial results. In other words, a specific solution is what is usually called a solution while a partial result is a part of a solution. A cluster solution is a set of features that the solution satisfies, and a partial cluster describes a part of a solution as a set of features.

An agent may generate proposals:

1. Independently, in which case, the agent generates proposals without taking into account other agents' proposals or critiques. Even in this independent case, the agent may make assumptions about the likely response of other agents in order to generate local proposals. These assumptions enable agents to operate concurrently even when dependencies exist, and if the assumptions are generally good, can speed the convergence process. Independent proposal generation occurs at the beginning of problem solving or when an agent is starting to negotiate an issue without any constraining information from others.

2. Based on proposals generated by other agents. Before submitting its proposal, the agent analyzes proposals made by the other agents. The submitted proposal may subsume, extend, or override some features of other agents' proposals.

3. Based on critiques made by other agents of previous proposals. In this case, the agent generates a new proposal which differs from its previous ones according to the other agents' critiques. Note that the critiques can concern its own proposals as well as other agents' proposals.

The structure and the content of a proposal are domain-dependent. The structure can include a *justification* that contains information about why and how the proposal was generated; this part is similar to *explanations* described in section 4.1.3.

4.1.2 Critiques

An agent evaluates a proposal formulated by another agent through a critique. The content of a critique is domain-dependent but the structure of the critique includes one or more of the following components:

1. A *positive* component that describes the features of the proposal the agent agrees with. It includes, for example, local constraints and goals that are supported by the proposal.

2. A *negative* component that describes the features of the proposal the agent disagrees with (the conflicting part of the proposal). It includes, for example, violated local constraints of the agent and global or local goals that are no longer achievable.

In addition to positive and negative components, a critique can be extended to include a justification and/or counterproposal. The content of the former is similar to the justification component introduced in the proposal structure and to explanations described in section 4.1.3. A counterproposal provides an alternative proposal that is more favorable to the responding agent. It has the same structure as the triggering proposal. The more information that can be provided in the critique (rather than just responding "OK" or "Not OK"), the better the receiving agent can understand and react appropriately.

4.1.3 Explanations

During negotiation, an agent often makes a proposal that is refused by another agent. This refusal may stem from incomplete or incorrect knowledge about the negotiated issue. In the case when an agent knows its knowledge is incomplete, before submitting a proposal or rejecting another agent's proposal based on assumptions, it can ask for more information that will likely lead to better proposals or analyses. Such information can have various forms, for example, data, partial hypotheses, goals, plans, or utilities. When an agent has incorrect knowledge or doesn't know that its local knowledge is not sufficient, responding agents may be able to provide explanations (or arguments) supporting their responses, i.e., information about the knowledge and reasoning used to evaluate the proposal. The agent that submitted the original proposal may then reason about whether or not to accept the received arguments and change its proposal in response.

4.1.4 Meta-Information

The use of meta-information is the least developed aspect of current negotiation protocols [20, 23, 24]. One role of meta-information in the negotiation process is to focus the local searches of agents so that potentially acceptable solutions or least-costly compromises can be found quickly. It can also play the role of indicating to agents whether a potential compromise is possible (likely) in the current situation, i.e., the current problem formulation is overconstrained, and which type of negotiation protocol is most effective to resolve the specific conflict. The information differs from critique information which is highly specific to the current proposal. Instead, meta-information concerns the general character of the agent's local search space (e.g., where likely compromise solutions can be found), and the agent's problem-solving capabilities (e.g., the type of critiques and explanations that agents can exploit).

4.2 Conflict Detection, Propagation, and Resolution

There are two ways in which problem solving may overlap among agents:

1. *Redundant goal achievement*: Multiple agents are working on the achievement of the same goal, and they make conflicting decisions regarding this goal. For instance, during the achievement of a common goal, the involved agents may produce conflicting results or syntheses of this goal.

2. *Interacting goal achievement*: Multiple agents are involved in the achievement of different but interacting goals, and there are constraining relationships among them. For instance, an agent's solution to its goal violates a local constraint of a second agent.

The definition of goals used here is more encompassing than domain problem-solving goals and includes meta-goals, such as determining the set of agents that will decide how to decompose a goal.

4.2.1 Conflict Detection

The negotiation process can start either explicitly, when the conflict results from an agent's critique of another agent's proposal, or implicitly, when separately generated proposals are inconsistent. Conflict detection can itself be a complex distributed problem-solving problem,

particularly in cases where agents are fully distributed and have no global view of problem solving [4]. In these situations, all information relevant to a problem must be explicitly communicated. Each agent's view of the current system state is potentially fragmented, it may be inconsistent with other agents' views, and it may be out-of-date. Recognition of a conflict may involve chains of inferences through multiple agents under uncertain conditions.

The conflict detection process can be:

1. *Outside* the scope of the agents, in which case all proposals, critiques, and arguments are sent to this external process.

2. *Centralized*, in which case a specific agent has a global view of the problem-solving process and is responsible for detecting all potential conflicts; the agent detects conflicts based on its view of the problem-solving state, on received proposals from other agents, or on previously detected ones.

3. *Distributed*, where a subset of the concerned agents are involved. In this case, an agent can detect conflicts that are:

   - directly specified in received information from other agents. The agent acquires knowledge about a conflict that has been detected by another agent. In other words, the latter tells the former that there is a conflict; this conflict may have been detected by a third agent.
   - deduced based on local activities. The agent detects conflicts independently of the work performed by the other agents. For instance, after performing some steps of local problem solving, the agent determines that the original problem cannot be solved without relaxing global constraints.
   - recognized based on received proposals/critiques or associated justifications. After evaluating received proposals or analyzing received critiques, the agent detects a conflict with its own perspective.
   - computed from other received or locally detected conflicts. For example, by propagating these conflicts through a network of constraints, the agent can determine other conflicts that, in turn, can be propagated to compute new ones and so on until all potential conflicts are detected.

   There can be an organization that maps each type of conflict to a set of agents that are responsible for detecting these conflicts. For instance, the detection of one type of conflict can be centralized in one agent while the detection of another type can be distributed among the entire community.

4.2.2 Conflict Propagation

Since multiple agents are involved in each negotiation, the need for propagation of a detected conflict arises: *To which agents does this conflict need to be signaled?* It may be that there are agents that will be affected by a particular conflict even though they are not directly involved in the resolution process. For example, two agents may conflict over the value of a parameter, x. One of these agents has a constraint, $x < y$. A third agent may know nothing about the existence of x but, nonetheless, will be influenced by negotiations over x because it is constrained on the value of y. This agent can make better local control decisions if it is aware of the conflict

(e.g., it may decide to defer decisions involving y until the conflict over x is resolved). The decision about where to propagate conflict information can be:

1. *Outside* the scope of the agents.
2. Determined by *the agent* that detected the conflict based on its local view of the problem-solving process.
3. Dynamically determined by one or multiple *fixed agents* based on the context of the conflict[6].

4.2.3 Conflict Resolution

Once a conflict or multiple conflicts are detected and possibly propagated, the issue of how to solve them arises. As a first response, one can imagine that this is the role of the agents that received the conflicts. But this is only one way to do it. In general, solving conflicts can be done by:

1. A *centralized* process where a specific agent is responsible for resolving all the conflicts among agents. This case assumes that the appropriate strategies to resolve conflicts and the criteria for stopping the negotiation process are held by this agent.
2. A *distributed* process where a subset of the concerned agents can be responsible for coordinating the negotiation process. In this case, the strategies for solving a conflict as well as the criteria for stopping the negotiation process can be either common or local to the agents. Generally, there is an organization that maps each type of conflict to a set of agents that are responsible for solving these conflicts.

Note that in a negotiation process, the agents that detect a conflict are not necessarily the same ones as those responsible for propagating the conflict. In the same manner, the agents to whom a conflict is signaled don't have to be the same as those that will be in charge of resolving the conflict. The role of the first ones can be limited to checking the consistency of their local constraints or to propagating the conflict.

4.3 Organizations

Once multiple agents decide to enter into a negotiation process, they may establish a common coordination strategy. Better coordination can lead to more coherent and efficient behavior through reduction of extraneous activity [1]. One way to establish a coordination strategy is to define an *organization* that provides a set of roles to focus the decision making of the involved agents. Since each negotiation involves a set of agents and one or more negotiation issues, there are, by necessity, dynamic coalitions in a negotiation-based CDPS system. In fact, there is exactly one organization per negotiated issue that specifies which agents formulate proposals and which agents evaluate these proposals. It also defines the agents responsible for detecting conflicts, propagating conflicts, solving conflicts, the agent that begins the negotiation process, the one in charge of controlling it, etc.

Unless it is predefined by the system designer or by an external process, defining an organization for a set of agents is a control problem-solving activity that involves one or more agents. In the latter case, this can also be done through a negotiation process.

[6] If the determining agents are not fixed, additional agents will be required to select them. This recursive process can go on indefinitely. We assume that at some level of recursion, these agents are predefined.

5 An Overview of Related Work

To understand the role of negotiation in existing systems, we examine a set of documented systems. These systems are representative of current research focusing on negotiation in CDPS, but are not intended to comprise an exhaustive list. Most work that has been done on negotiation for CDPS primarily investigates issues that occur when multiple agents are achieving interacting subproblems. However, there are exceptions to this and each project has a different focus. The term *negotiation* is not well-defined across the set of projects discussed here: each researcher uses it in a slightly different way. Therefore, we will differentiate between negotiation, meaning communication of information for the purpose of finding a mutually acceptable solution, and conflict resolution, meaning direct application of some technique designed to eliminate a conflict that has occurred.

Cammarata et al.: Cammarata and her colleagues [2] applied negotiation in the domain of distributed air-traffic control. Each airplane (agent) in this domain constructs a flight plan that maintains an appropriate separation from other airplanes and satisfies other constraints such as getting to the desired destination with minimum fuel consumption. Their most well-developed approach to this problem is a policy called task centralization. In this policy, airplanes involved in potential conflict situations (which occur when airplanes could get too close, based on their current headings) choose one of the airplanes involved in the conflict in order to resolve it.

In this work, the focus is on allocating the task of conflict resolution to the agent who has the best knowledge to determine the action to take to resolve the conflict, and has the capability for taking such action. Although multiple agents are involved in this selection process, no conflict exists because all aircraft have the same information and criteria for making selections. Further, because it is assumed that a single agent can modify its plans to resolve a conflict, sophisticated, distributed negotiation strategies among agents are not needed.

Davis and Smith: Davis and Smith [5] studied the use of negotiation for the allocation of tasks in their Contract Net Protocol[7]. At each stage, a task can be directly achieved by an agent without interaction with other agents or by being decomposed into subtasks and allocated. In the latter case, the agent interacts with the other agents in order to decide where to transfer its subtasks; this is accomplished by exchanging with them task announcements, bids, and awards. They reach agreement when the manager (which requested bids for solving one of its subtasks) awards a contract to a bidder (which proposed bids for solving this subtask).

In this approach, initial tasks that characterize the problem are predefined and allocated to an agent. A task is directly achieved by the agent to which it is allocated or it is further decomposed by that agent. Agents play specific hierarchical roles in the allocation process, with the allocating agent being the manager and the agents that could potentially achieve a task being bidders. A task is active when its manager decides to allocate it to other agents. Task allocation is decided by the agents through a single-shot negotiation process consisting of a mutual selection between the manager and a potential contractor. It should be noted that conflicts arise when multiple agents simultaneously bid on the same task. Conflict resolution is always hierarchical; it occurs between the manager and each of the bidders but not among the bidders.

[7] Further work on this protocol has been done by Van Dyke Parunak for use in a manufacturing environment [24].

The work of Cammarata et al., and Smith and Davis, though historically seen as the earliest uses of negotiation in CDPS, are extremely elementary forms of negotiation from the perspective of the RNM framework. In fact, there is some validity to the viewpoint that they don't really represent the use of negotiation. Each can be characterized as a single-shot process involving a selection among alternatives[8]. In neither of these systems is there any reconciliation of disparate viewpoints or relaxing of local agent requirements on a subproblem to achieve an acceptable overall solution. However, a variant of the Contract Net Protocol used by Durfee and Lesser for task allocation in the Partial Global Planning framework [9] does in fact represent a process that is more closely aligned to our view of negotiation. Their protocol is a multi-step process where the bidder's response to the contractor's task announcement can cause the contractor to reorganize its task announcement to be more aligned to the bidder's response. This cycle can then continue until a consensus is reached between the contractor and the bidder or a decision is made to break off the negotiation.

Klein: Klein [15, 16] proposes that conflict resolution expertise exists separately from domain-level expertise. He has developed a system, *Cooperative Design Engine* (CDE), based on a taxonomy of conflict classes and associated general advice for resolving conflicts in each class. Each agent in CDE has different domain expertise and the same conflict resolution expertise. Conflicts occur through interacting subproblems. The application domain described is local area network design, a resource management problem. Problem formulation, task allocation, and focus-of-attention are outside the scope of the agents. Problem reorganization is possible at intermediate levels of the problem structure.

Klein's application of negotiation has been limited to the domain of problem solving and does not focus on its use for control and organizational design problem solving. However, to be fair to his work, it could be argued that his basic approach to conflict resolution can be extended to handle these other uses of negotiation. Another aspect of his work that seems limited is his view of conflict resolution as a single-shot process, where information is not gained from each cycle of conflict resolution; there is also no explicit notion of mutual accommodation of the agents' perspectives.

Conry, Kuwabara, Lesser, and Meyer: They develop a multi-stage negotiation protocol [3, 4, 18] for detection and resolution of resource allocation conflicts in a distributed network. The focus of this work is to enable agents, while searching for an acceptable solution, to also detect overconstrained situations. Conflicts occur among agents due to local and non-local resource allocation constraints and due to an inconsistent view of problem solving. Negotiation involves sharing information to guide the search for a consistent viewpoint. Their asynchronous and distributed protocol involves determining the impact of local decisions on non-local decisions by allowing each agent to exchange information about 1) its local decisions about resource allocation; 2) the effect of a local decision on other potential goals/plans that could be pursued; and 3) information received from other agents. Goals are pre-allocated to agents. The choice of local solutions during negotiation implicitly defines which goals are active. When an overconstrained situation is recognized, agents can choose appropriate goal relaxations based on shared information and evaluation criteria.

This multistage negotiation protocol is structured into three phases: an asynchronous search

[8]This is not a totally accurate presentation of the Contract Net Protocol since a contract can be constructed from parts of bids of multiple agents.

phase, a coordinated search phase, and an overconstrained resolution phase. Each of these phases requires agents to handle goals in a progressively more coordinated way, based on aggregated evidence accumulated about the interrelationship among solutions to goals. In the asynchronous search phase, which occurs first, agents try to find solutions independently of other agents. The coordinated search phase is entered when all the choices of an agent with respect to its local goals have been tried and an acceptable solution has not been found. In this case, it may be necessary to retry alternative solutions that have previously failed due to their incompatibility with tentative solutions to non-local goals created by other agents. This is necessary because other agents, in the meantime, may have retracted their solutions to these non-local goals. This phase of the search requires coordination among the agents to guarantee that all combinations of solutions to goals are examined for their feasibility. If a solution to all goals is not found in this phase an overconstrained situation has occurred. In this case, the overconstrained resolution phase decides which goals need to be abandoned in order to resolve the situation.

This work shows some of the complexities that occur when agents asynchronously negotiate about the solutions to interacting subproblems when a global view is lacking. Their approach integrates the processes of conflict detection, recognition of an overconstrained problem, and negotiation into a single protocol which simultaneously accomplishes all three activities. This work is limited, however, in that it does not take into consideration that some solutions are preferable to others. For example, some solutions may be preferable in terms of what resources they require. It is also limited since it only uses simple constraints among subproblems based on the location and availability of resources.

Lander and Lesser: Lander and Lesser [19, 20] describe a framework, TEAM, for multi-agent cooperative design of mechanical systems using *negotiated search*. This work focuses on integrating distributed search among a set of heterogeneous and reusable agents. The agents in the agent set may be representationally heterogeneous, having different architectures (e.g., a neural net and a rule-based system), inference methodologies (e.g., backward- or forward-chaining), or languages; or the agents may be logically heterogeneous with different knowledge bases, goals, priorities, or evaluation criteria. An agent is reusable if it is designed explicitly so that it can be used in any agent set without *a priori* knowledge of the overall application. For example, an agent that designs a particular type of component can do so no matter what system that component will be embedded in. Heterogeneity and reusability require a great deal of local agent autonomy and sophistication. Each agent must be given an opportunity to evaluate any proposed solution since it may be the only agent that has some key piece of information about that solution. Each agent's evaluation must be respected in considering the acceptability of a solution. An agent cannot rely on models of other agents to predict the actions of others or to coordinate its local behavior with others. Conflict is the primary stimulus for controlling agent interaction and coordination.

In TEAM, negotiation is initiated through the detection of a conflict as an agent critiques or extends a proposal initiated by another. The detecting agent notifies the initiating agent of the conflict and, to whatever extent possible, communicates relevant information about the cause of the conflict. The initiating agent gathers critiques of its proposal and integrates the shared information into its own knowledge base, making choices about how best to do that. Those choices can include relaxing its own solution requirements to accommodate another agent or refusing to relax a particular requirement even though it is explicitly incompatible with a requirement stated by another agent. In consequence, the final proposed solution may be unac-

ceptable to one or more agents under the complete set of preferences. Each agent participating in the negotiation has the option of accepting or rejecting a proposed solution. If a solution is initially rejected, it remains available and may be reconsidered in the future as agents continue to relax requirements in their search for a mutually-acceptable solution.

The TEAM framework provides a flexible control structure that allows agents to share information about their problem-solving capabilities and preferences with respect to a particular problem. This information is used to dynamically select, instantiate, and execute the most effective conflict resolution strategy possible given the local requirements of each agent in a specific agent set and the application problem. In this way, the agents use negotiation both for distributed meta-level control of the system and for distributed domain-level problem-solving control.

Moehlman and Lesser: Moehlman and Lesser [22, 23] describe a framework, called DENEGOT, for negotiating conflicts that arise in multi-agent planning with time and cost constraints. Top-level goals are originally predefined with some threshold level of global cost and time utility required. Agents own resources and have predefined responsibilities. Goal achievement is distributed. If an agent cannot find a plan to optimally meet the top-level goals it is responsible for with its own resources, the agent negotiates with other agents to borrow resources to help achieve its goals. This results in an exchange of proposals and critiques among the agents. The intent of negotiation is to find a combined multi-agent plan in which all the top-level goals are satisfied in an acceptable, though not necessarily optimal, fashion. If this negotiation process fails, the agents then enter a meta-level negotiation process to decide how to lower the criterion of acceptability of the top-level goals in order to find acceptable solutions for all the goals. As part of the negotiation process, agents can exchange meta-level information indicating likely places for compromise.

They view this negotiation process as a satisficing distributed search. This involves conceptually partitioning the composite search space of agents into a lattice of sets of potential compromise solutions. A solution in a higher set in the lattice, if it is achievable, will be preferable over a solution in a lower set. Solutions within a set may be further ordered in terms of their preference. This conceptual partitioning of the composite search space is implemented in terms of partitions of each agent's local search space that mirror the structuring of the composite search space. The distributed search for a solution is based on finding the highest set in the lattice in which a compromise solution acceptable to all agents can be constructed. Focusing the search on a set lower in the lattice can be thought of as making compromises in order to find a mutually acceptable solution. This distributed search is implemented as three iterative problem-solving phases: coordinated search, negotiation state analysis, and constraint relaxation.

von Martial: Von Martial's [32] work on negotiation relates to coordinating and synchronizing pre-formed plans generated locally by autonomous agents. Negotiation is used both to resolve conflicts caused by shared access to non-consumable resources by the plans of agents, and to exploit the potential for one agent to slightly alter its plan in such a way that the number of actions required by another agent to carry out the intended purpose of its plan is reduced. In a principled way, the alternative strategies for resolving conflict are detailed based on temporal ordering constraints; additionally, heuristics are specified for choosing the order of resolving multiple conflicts among agents and for choosing among alternative strategies for resolving conflicts. Quantitative heuristics are also given for deciding during negotiation when and how

agents should restructure their plans to exploit beneficial synergy.

The distributed negotiation protocols used by agents to recognize and resolve/exploit plan interactions are some of the most sophisticated strategies yet devised for coordinating agent activities. They exploit the hierarchical nature of plan descriptions to reduce communication among agents, and can work with only partially expanded plan networks. These complex, multi-step, asynchronous protocols are extremely hard to verify for correctness. One of the innovations of this work is the application and reworking of formal verification techniques, developed for the analysis of low-level network communication protocols, so that they can be used to verify these high-level negotiation protocols.

Sathi: Sathi [28] describes a negotiation framework based on constraint relaxation for resolving conflicts in resource reallocation. He defines a set of negotiation operators that an agent may use to propose a compromise:

- *Log-rolling*, where each agent slightly relaxes its interacting constraints
- *Reducing the cost* of a relaxation by agreeing to relax the solution criteria
- *Substitution* of a less preferred resource in place of the preferred resource
- *Bridging*, which involves the development of a completely new solution that satisfies only the most important constraints
- *Unlinking*, which involves overlooking weak interactions among constraints
- *Mediation and arbitration*, where third parties that possess additional knowledge and/or authority are drawn into the negotiation process

Control of the problem-solving process is through the central framework. However, each agent computes utilities on partial solutions locally. The focus of this work is on using a constraint-based representation in a negotiation framework where partial solutions are incrementally extended, constraint relaxation is applied appropriately, and creative solutions can be developed. Sathi uses domain knowledge in an interesting way to physically change objects in the environment to meet resource demands.

Sycara: Sycara [31] built a centralized planner, *PERSUADER*, for resolving conflicts between agents' goals. Given a particular conflict and the context in which the conflict occurs, the planner has two alternatives: (1) find a new compromise by using case-based reasoning or multi-attribute preference analysis strategies, or, (2) use persuasive arguments to convince agents to accept the proposed compromise by using explanation-based reasoning or try to improve the compromise by asking for justification of disagreements. Her techniques for forming compromises and for maintaining a history of previous negotiations and compromises are useful for taking advantage of the experience the system has acquired in negotiation. In this approach, the formulation of top-level goals is predefined and allocated to agents. All common goals are active at each step of the problem-solving process and the agents that will participate in solving these goals through negotiation are predefined. Since conflicts are given as a part of the original problem, there is no conflict detection mechanism.

Negotiation is coordinated by a centralized mediator, making it an explicit process that is outside the scope of the agents. The mediator has access to local and global information about

the situation and about the negotiation participants. Mediation is particularly useful when the parties involved in a conflict are antagonistic. If the disputants are dishonest or hostile, a mediator provides a buffer zone in which information can be collected and evaluated. However, mediation is a costly process because all necessary local information must be explicitly collected from the disputants or calculated by the mediator. Furthermore, the mediator must know enough about all aspects of the domain to make intelligent decisions.

A potential criticism of using negotiation in CDPS is that it can be a costly and time-consuming process and, consequently, it may increase the overhead of problem solving [17]. *Case-based reasoning* [26, 30] is a possible approach to reducing the overhead of negotiation. Sycara used case-based reasoning to avoid negotiating a conflict from scratch by retrieving and refining solutions to similar cases.

Werkman: Werkman [33] has developed the *Designer Fabricator Interpreter* (DFI) that attempts to reconcile the different perspectives of designer, fabricator, and erector agents to choose connections for building design. DFI is supported by a knowledge representation language that links objects in the domain to the alternative agent perspectives. The negotiation process is facilitated by an arbitration agent that, for the most part, centralizes system control. Problem formulation and decomposition are implicit in the system. The focus in this system is on resolving conflicts that arise from the conflicting evaluation criteria of different agents, that is, each agent is working on the same problem, but taking a different perspective. This centralized negotiation process, which assumes knowledge of the agents' criteria, results in a pair-wise mutual accommodation of the viewpoints of agents.

Human Negotiation and Creativity: Pruitt [25] discusses human negotiation and conflict and defines a set of negotiation strategies. His work comes from a background of social psychology and is grounded in domains such as international relations. Much work in human negotiation deals with psychological issues such as how to make concessions without losing power. Although many insights can be gained from this work, most psychological factors are not immediately relevant to computational negotiation.

A prominent feature of intelligent-agent problem solving is that solutions are sometimes unanticipated or unpredictable. As pointed out by Hewitt [14], the outcome of negotiation among humans is often unintended by the participants and may be undesirable to some participants. DeBono [6] defines *lateral thinking* as a mechanism for generating innovative solutions in human problem solving. Lateral thinking is a method in which different approaches to a problem are explicitly sought through strategies which are designed to elicit unusual connections between problem elements. Although DeBono does not speak directly to the issue of negotiation of conflicts, his work offers insight into the type of creative problem solving that can be used during search to form innovative solutions. RNM is a particularly suitable model for creative problem solving since it supports the exchange of unpredictable information among possibly heterogeneous agents, each with a local perspective on a shared solution.

Game Theoretic Approaches to Negotiation: Georgeff [11] discusses a model for solving a restricted class of multi-agent problems in which agents are able to cooperate without explicit communication. The model assumes that each agent has complete knowledge of all other agents' utilities and payoffs. This limitation is too severe for many real-world problems. One of the compelling reasons for developing an RNM is to allow agents to negotiate solutions when they don't

have good models of each others' utilities and payoffs. Several researchers have extended formal game theoretic approaches to more generalized models for negotiation in multi-agent problems where the agents do communicate, where information may not be complete, and where agents may not be truthful in their communications [12, 13, 27, 34, 35]. However, these models still assume the ability to calculate the payoff matrices for all agents and the ability to compare potential payoffs for all alternative solutions. For this reason, the models cannot be applied in complex problem-solving situations. One of the goals of game theory models is to explore the utility of different search techniques formally. This analysis may become a valuable tool for developing heuristics in situations where it is not possible to exhaustively explore the set of potential solutions.

Discussion

The approaches to negotiation outlined above generally deal with conflicts that occur during one stage of problem solving or that occur due to a specific pattern of interaction. For example, the Contract Net Protocol uses negotiation to resolve conflicts in the allocation of active goals to agents. TEAM uses distributed negotiation operators to detect and resolve conflicts in interacting subproblem solutions and negotiation strategies to coordinate the application of particular operators. PERSUADER uses a centrally coordinated set of negotiation operators to resolve externally specified conflicts that occur due to antagonistic perspectives on the value of a solution. Additionally, most of these existing systems implement negotiation as a centralized process that is not an integral part of domain problem solving. Thus, many of the potential uses of negotiation in CDPS as described in the RNM model have yet to be explored.

6 Conclusions

The development of the Recursive Negotiation Model came from our desire to understand the role of negotiation in sophisticated CDPS systems. As a result of building this model, we have shown not only how pervasive the role of negotiation in a CDPS system can be, in terms of domain, control and meta-level control (organizational) problem solving, but we have also provided the reader with a framework for comparing and contrasting different approaches to negotiation. The most notable aspects of this model are: 1) its integration of the task- and results-sharing model for CDPS into a unified framework; 2) its clear delineation of the four main stages of CDPS problem solving, in terms of this integrated model for CDPS (goal formulation, focus-of-attention to select active goals, allocation of goals to agents, and achievement of goals) and the flow of control among the stages; 3) its recognition of the role of negotiation and conflict detection as a complex, multi-level and recursive process, which can occur at all stages of CDPS problem solving; and, finally, 4) its model for the key information exchanged among agents based on viewing negotiation in terms of exchange of proposals, critiques, explanations, and meta-information. We see this work as a first step towards developing a high-level architectural framework in which CDPS systems can be specified and analyzed. By explicitly defining potential roles for negotiation within each stage of problem solving, we provide a model that can be flexibly tailored to the problem at hand.

Acknowledgements

This research was done at the University of Massachusetts at Amherst and was supported in part by the Office of Naval Research (ONR) under contract number N00014-89-J-1877, by the French National Institute for Research in Computer Science and Automation (INRIA), and by the French Lorraine County.

We would like to thank Keith Decker, Norman Carver, Daniel Corkill, and Dorothy Mammen for their comments. Thanks also to Michele Roberts and Stephan Brunessaux for their editorial comments.

References

[1] A. Bond and L. Gasser, editors, *Reading in Distributed Artificial Intelligence*, Morgan Kaufmann Publishers, 1988.

[2] S. Cammarata, D. McArthur, and R. Steeb, "Strategies of Cooperation in Distributed Problem Solving," *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, Karlsruhe, West Germany, 1983, pp. 767-770.

[3] S. E. Conry, R. A. Meyer, and V. R. Lesser, "Multi-Stage Negotiation in Distributed Planning," in A. Bond and L. Gasser, editors, *Reading in Distributed Artificial Intelligence*, Morgan Kaufmann Publishers, 1988, pp. 367-384.

[4] S. E. Conry, Kuwabara, K., Lesser, V. R. and Meyer, R. A. "Multistage Negotiation in Distributed Constraint Satisfaction," *IEEE Transactions on Systems, Man and Cybernetics* – Special issue on DAI, Vol. 21, No. 6, November/December 1991, pp. 1462–1477.

[5] R. Davis and R. G. Smith, "Negotiation As a Metaphor for Distributed Problem Solving," *Artificial Intelligence*, Vol. 20, 1983, pp. 63-109.

[6] E. de Bono, *Lateral Thinking for Management, A handbook of creativity*, American Management Association, 1971.

[7] K. S. Decker and V. R. Lesser, "Some Initial Thoughts on a Generic Architecture for CDPS Network Control," *Proceedings of the Ninth Workshop on Distributed Artificial Intelligence*, Rosario Resort, Eastsound, Washington, September 12-14, 1989, pp. 73-94.

[8] E. H. Durfee and V. R. Lesser, "Using Partial Global Plans to Coordinate Distributed Problem Solvers," *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, Italy, 1987, pp. 875-883.

[9] E. H. Durfee and V. R. Lesser, "Negotiating Task Decomposition and Allocation Using Partial Global Planning," in L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, Vol. 2, Pitman, 1989, pp. 229-243.

[10] E. H. Durfee, V. R. Lesser, and D. D. Corkill, "Cooperative Distributed Problem Solving," in A. Barr, P. R. Cohen, and E. A. Feigenbaum, editors, *The Handbook of Artificial Intelligence*, Vol. 4, Addison Wesley, 1989, pp. 83–147.

[11] M. Georgeff, "Communication and Interaction in Multi-Agent Planning," *Proceedings of the National Conference on Artificial Intelligence*, Washington, DC, 1983, pp. 125–129.

[12] P.J. Gmytrasiewicz, E.H. Durfee, and D.K. Wehe, "A Decision-Theoretic Approach to Coordinating Multiagent Interactions," in *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, August 1991.

[13] P.J. Gmytrasiewicz, E.H. Durfee, and D.K. Wehe, "The Utility of Communication in Coordinating Intelligent Agents," in *Proceedings of the National Conference on Artificial Intelligence*, July 1991, pp. 166–172.

[14] C. Hewitt, "Towards Open Information Systems Semantics," *Proceedings of the Tenth International Workshop on Distributed Artificial Intelligence*, Bandera, Texas, October 23–27, 1990.

[15] M. Klein, "Supporting Conflict Resolution in Cooperative Design Systems," *Proceedings of the Tenth International Workshop on Distributed Artificial Intelligence*, Bandera, Texas, October 23–27, 1990.

[16] M. Klein, "Supporting Conflict Resolution in Cooperative Design Systems," *IEEE Transactions on Systems, Man and Cybernetics* – Special issue on DAI, Vol. 21, No. 6, November/December 1991, pp. 1379–1390.

[17] S. Kraus and J. Wilkenfeld, "The Function of Time in Cooperative Negotiations," *Proceedings of the Ninth National Conference on Artificial Intelligence*, Anaheim, California, July 14–19, 1991, pp. 179–184.

[18] K. Kuwabara and V. R. Lesser, "Extended Protocol for Multi-Stage Negotiation," in *Proceedings of the Ninth Workshop on Distributed Artificial Intelligence*, Rosario Resort, Eastsound, Washington, September 12–14, 1989, pp. 129–161.

[19] S.E. Lander, V.R. Lesser, and M.E. Connell. Knowledge-Based Conflict Resolution for Cooperation among Expert Agents. In D. Sriram, R. Logher, and S. Fukuda, editors, *Computer-Aided Cooperative Product Development*, Springer-Verlag, 1991, pp. 253–268.

[20] S.E. Lander and V.R. Lesser. Negotiated Search: A Framework for Cooperative Design. Technical Report 91–79, Department of Computer and Information Science, University of Massachusetts, Amherst, Massachusetts 01003, November 1991.

[21] V.R. Lesser, "A Retrospective View of FA/C Distributed Problem Solving," *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1347–1362, Nov/Dec 1991.

[22] T. Moehlman and V.R. Lesser. "Cooperative Planning and Decentralized Negotiation in Multi- Fireboss Phoenix," in *Proceedings of the 1990 DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control*, Texas, November 1990.

[23] T.A. Moehlman, V.R. Lesser and B.L. Buteau, "Decentralized Negotiation: An Approach to the Distributed Planning Problem," *Group Decision and Negotiation*, 1:2, pp. 161–192, Kluwer, 1992 (to appear).

[24] H. Van Dyke Parunak, "Manufacturing Experience with the Contract Net," in *Distributed Artificial Intelligence*, M. Huhns (ed.), Pitman/Morgan Kaufmann Publishers, pp. 285–310.

[25] D. G. Pruitt, *Negotiation Behavior*, Academic Press, New York, 1981.

[26] C.K. Riesbeck and R.C. Shank, "Inside Case-Based Reasoning," Lawrence Erlbaum Associates, Publishers, 1989.

[27] J.S. Rosenschein, "Deals Among Rational Agents," *Proceedings of the International Joint Conference on Artificial Intelligence*, 1985.

[28] A. Sathi and M.S. Fox. "Constraint-Directed Negotiation of Resource Reallocations," in Les Gasser and Michael Huhns, editors, *Distributed Artificial Intelligence, Volume 2*, Pitman/Morgan Kaufmann Publishers, 1989, pp. 163–193.

[29] R. G. Smith and R. Davis, "Frameworks for Cooperation in Distributed Problem Solving," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 11, No. 1, 1981, pp. 61–70.

[30] K. Sycara, "Arguments of Persuasion in Labour Mediation," *Proceedings of the International Joint Conference on Artificial Intelligence*, 1985, pp. 294–296.

[31] K. Sycara, "Resolving Goal Conflicts via Negotiation," *Proceedings of the Seventh National Conference on Artificial Intelligence*, St. Paul, Minnesota, August 1988, pp. 245–250.

[32] F. von Martial, *Lecture Notes in Artificial Intelligence*, Springer-Verlag, 1992.

[33] K.J. Werkman. *Multiagent Cooperative Problem-Solving through Negotiation and Sharing of Perspectives.* PhD thesis, Lehigh University, May 1990.

[34] G. Zlotkin and J. Rosenschein, "Negotiation and Conflict Resolution in Non-Cooperative Domains," *Proceedings of the Eighth National Conference on Artificial Intelligence*, Boston, MA, July 1990, pp. 100–105.

[35] G. Zlotkin and J.S. Rosenschein, "Cooperation and Conflict Resolution via Negotiation Among Autonomous Agents in Non-Cooperative Domains," *IEEE Transactions on Systems, Man, and Cybernetics*, Special Issue on DAI, Vol. 21, No. 6, November/December 1991, pp. 1317–1324.