A Proposal Toward Quantified, Organizationally Centered, Decision Making and Coordination for Software Agents

Thomas Wagner and Victor Lesser
wagner@cs.umass.edu

UMASS Computer Science Technical Report 1998-83

December 1, 1998

Abstract

To scale agent technologies for widespread use in open systems agents must have an understanding of the organizational context in which they operate. The focus of this research is on the expansion of agent knowledge structures to support modeling of organizational information and on a corresponding expansion of agent control techniques to use the information. Of particular interest is the issue of task valuation and action selection in such socially situated agents – specifically on the issue of quantifying agent relationships and relating work motivated by different sources. For example, the comparison of work done for self-interested reasons to work motivated by cooperative strategies.

This document is a portion of a Ph.D. proposal. It is being published separately as several of the intellectual ideas contained in this document were trimmed from the doctoral research. Please consult the dissertation for more meaningful research.

A Proposal Toward Quantified, Organizationally Centered, Decision Making and Coordination for Software Agents

Thomas Wagner and Victor Lesser

December 1, 1998

Abstract

This document is a portion of a Ph.D. proposal. It is being published separately as several of the intellectual ideas contained in this document were trimmed from the doctoral research. Please consult the dissertation for more meaningful research.

In order to scale-up agent technology for use in open application domains, e.g., electronic commerce on the web, agents must model their organizational relationships with other agents and reason about the value or utility of interacting and coordinating with particular agents. For example, a database management agent owned and operated by IBM might have an extremely cooperative relationship with an information gathering agent owned by Lotus (Lotus is a subsidiary of IBM), but an entirely different type of relationship with a Microsoft information gathering agent – the IBM agent might prefer to service requests for the Lotus agent over the Microsoft agent or it might be willing to cooperate with the Microsoft agent if a higher fee is paid for its services. The agents might even coordinate via different protocols; the IBM agent might haggle with the Microsoft agent over delivery time and price whereas it might simply satisfy the Lotus request in short order and with a nominal or zero profit margin.

For agents to act rationally in light of their different relationships, agents must be able to reason about their different organizational roles and decide which actions to perform, and when, from a worth or utility driven perspective. As agents have finite resources (bounded rationality), the ability to evaluate candidate actions in this light is critical. This research differs from work in multi-agent market mechanisms in that it deals with the temporal scope and sequencing of actions, not the market determination of the prices for goods and services. This research also differs from temporal coordination work (e.g., GPGP) as the focus is not on the dialogue held by agents to exchange temporal constraints per se, but on the determination of which actions to perform, with whom, and when. This research is complementary to both of these two areas of multi-agent research as it is centered on the local agent's in-context valuation of its candidate actions, where the actions may include coordination actions (those actions that pertain to carrying on a dialogue with another agent). In a sense, the valuation process *motivates* coordination, e.g., GPGP, and the coordination processes may employ market mechanisms to arrive at candidate prices.

This research proposes the development of knowledge structures for representing organizational knowledge (including organizational roles), joint goals, commitments between agents, and primitive actions. The new and enhanced structures must then be combined quantitatively via an interrelated web of influences that is reasoned about by a new decision process component. The decision process is responsible for determining which actions to perform and when, based on predictions of action outcomes and the web of influences between the different artifacts. The decision process must take into account task interactions, and constraints such as cost limits or the temporal constraints of other agents, e.g., deadlines, when performing this valuation process. The temporal in-context evaluation of actions is necessary as very attractive actions might be completely unachievable – possibly resulting in chains of interaction effects. Since relationships may evolve over time, and agents may produce new goals or tasks (possibly by interacting with other agents), the context in which the value or utility of actions is determined is also dynamic. Consequently, the decision process must be efficient and able to adapt to the changing context and changing knowledge structures.

Contents

Chapter 1

Forward

This document is a portion of a Ph.D. proposal. It is being published separately as several of the intellectual ideas contained in this document were trimmed from the doctoral research. Please consult the dissertation for more meaningful research.

Chapter 2

Introduction

In this chapter we introduce the thesis topic assuming a working knowledge of multi-agent systems and the state of the art of multi-agent systems. Prior to delving into the topic directly, readers unfamiliar with multi-agent systems may want to skip ahead and read Chapter 3, which provides background and places the proposed research in the context of the larger multi-agent systems community. In Chapter 4 we discuss the current state of our local research (as the thesis builds on these ideas) and in Chapter 5 we discuss the thesis topic in detail. Related work is referenced and discussed throughout the document, though Chapter 6 discusses particularly relevant items in detail. Chapter 7 sketches work and experimentation plans.

2.1 Introduction to Wide Context Decision Making

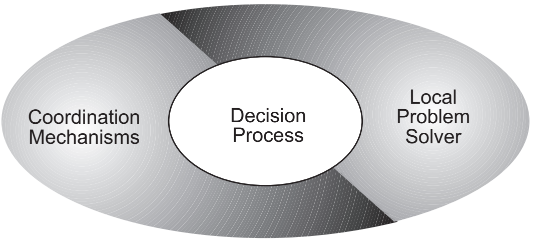


Figure 2.1: The Decision Process is the Heart of Agent Control

The main focus of this thesis is to extend agent knowledge structures and analysis procedures to include notions of organizational structure and knowledge. This, in turn, will enable more complex structuring of groups of agents and will facilitate coordination protocols for dealing with larger groups of agents. For example, behaviors like collective bargaining[1] or the formation of coalitions and alliances could explored. Currently, little work in multi-agent systems

[1] This research will not deal with the model of treating groups of agents as a single agent from a coordination perspective. However, the relationship models developed in this work will support protocols of this type by enabling agents to defer strongly to the will of another agent. This can be used to elect or appoint a leader for a group of agents or to give a particular agent authority to negotiate with other agents on the behalf of the group. Depending on the grainsize of the model, this may require other support, e.g., parallel scheduling abilities, which is beyond the scope of this work. The support provided by this proposed research is in the expression of *relationships* between agents and they way in which they influence or affect the choices that a given agent will make.

addresses the social context of the agents. Most work instead focuses on coordinating the activities of small groups of agents. In order to scale up agent technology for widespread use, agents need to be able to reason about their roles from an organizational perspective – agents must decide what to do, when, and with whom by understanding the relative importance of their various candidate activities and part of the importance is dependent on the organizational relationships with which the actions are associated. In general, the proposed knowledge extensions change the agent control equation by providing a richer context from which agents can make control decisions. For example, the new structures will represent situations such as: agent β is α's direct superior, and γ is a peer of β but not α's direct superior; thus if γ requests that α perform some task, and the task is in conflict with an equally high priority task for β, α can reason about the relative importance of the tasks and select β's task accordingly. The ability to evaluate different options from a value or worth-driven perspective is important not only because it enables agents to explicitly evaluate their relationships, but also because it increases agent flexibility; through this view, agents can respond appropriately to changes in the environment, in the agent system, in their own goals, and so forth. The new knowledge structures enter into the agent control equation in two primary ways:

Quantitative Decision Making The action-selection-sequencing problem is one of the central aspects of agent control and coordination. Generally, agency (Chapter 3) implies having multiple different goals to achieve and possibly multiple alternative ways to achieve them. Thus, agents must *select which* actions to perform and *choose when* to perform them. Henceforth we will refer to this activity, the process of selecting a course of action from a set of candidate actions, as the local agent *decision process*. In our work and indeed in the "real world," different actions have different statistical characteristics or different performance profiles, thus to achieve *rational* [71, 72] behavior the decision process must factor-in the different relative values of candidate actions and select an overall course of action that maximizes utility (defined by some objective function or complex criteria).

The valuation of the candidate actions in this process is contextually dependent. For example, if time is limited, an activity that produces a poor quality result, but does so very quickly, may be preferred over an activity that produces a high quality result but requires a very long time in which to produce the result. The proposed quantified, organizationally centered knowledge structures expand this context by specifying notions like quantified power relationships between agents, quantified preference relationships or avoidance relationships, and so forth. The information also specifies whether agents interact at the self-interested end of the coordination spectrum or at the more altruistic end. This quantified information must be incorporated into the local agent decision process and factored into the action-selection-sequencing problem. In other words, the new information expands the context in which the value or utility of primitive actions is determined. This, in turn, will enable agents to reason about the relative costs and benefits of interacting with particular agents over particular tasks.

It is important to note that the decision process is ongoing. As the environment changes (e.g., new alliances are formed, resources become more scarce, etc.) and the state of problem solving evolves, the context changes and the relative importance or value, and relative costs, of particular actions change. Thus the decision process is repeated throughout an agent's existence (though there may be islands of stability within this landscape). The dynamic character of the evaluation is important for two reasons: 1) the recognition that as the context changes, so too may the valuation of candidate actions (and committed courses of action) and thus the process must be repeated, and 2) the decision process component must be able to initiate activity, e.g., initiate coordination to obtain more information about what is happening at other agents or initiate its own analysis process. Implementationally this might be performed by a separate agent control component that allocates time to the decision process, coordination module, and other agent components. This pro-active behavior might also be achieved by describing triggers for another component that cause the decision process to be notified when particular events occur.

Distributed Computation Structure In addition to providing new quantitative context for action evaluation and selection, the new knowledge also specifies attributes that structure agent interactions. The types of structural elements contained in the organizational knowledge include: 1) which agents are likely to interact, 2) how they are likely to coordinate or which protocol to use[2], 3) the tasks over which they are likely to coordinate, 4) how agents of one group relate to agents of another group, etc.; all of which impose new structure on agent coordination. Information of this class can be incorporated into agent coordination protocols to reduce the amount of communication necessary to coordinate agents and to bias or predispose agents toward certain behaviors. This will enable MAS builders to construct intricate networks of agents without facing a combinatorial explosion of the coordination problem – organizing the distributed computation will enable the agents to coordinate and communicate with a limited number of other agents, regardless of how large the system may be. Additionally, the structural information replaces or supplements communication-based coordination – agents do not need to broadcast to all the other known agents to detect which agents are part of the same organization or structure of agents and so forth. The thesis contribution in this area will be the design and support of knowledge structures to represent such concerns. The full utilization of these structures is an area of future research. In this thesis we will focus on the quantitative, decision process aspect of the agent control equation.

We make the distinction between the two different uses of the information because they pertain to different aspects of agent control in multi-agent systems. Information that influences[3] the relative, contextually dependent, value of candidate actions (domain actions, coordination actions, communication actions) pertains mostly to the local agent decision process. This change does affect coordination activities, though indirectly, because it ultimately determines which tasks and actions an agent will perform. Historically we have only evaluated domain problem solving actions, in this new work we may also value and reason about coordination and communication actions. Thus, the new information affects the coordination in two ways: 1) by determining which domain actions an agent will perform and thus determining which interactions may be coordinated, and 2) possibly by determining which coordination and communication actions will be performed and which ones are not *worth* performing. These effects are *indirect* in that they do not change the coordination mechanisms (e.g., a mechanism to coordinate a hard precedence relationship) directly, but instead influence the cases in which the mechanism is employed. Obviously, it is a different matter if new organizationally centered coordination protocols (e.g., collective bargaining, coalition formation, buying syndicates) are developed to leverage the new information.

In contrast, information such as which agents are likely to interact with a given agent, or which protocols the agent should use, pertain mainly to the structure of the distributed computation and thus relates to the coordination process directly. Much of this information is not directly relevant to the local agent decision process but takes the role of parameters or bindings on the coordination protocols. The effects of this type of information on the local decision process are mainly in terms of the characterization of coordination actions and in the *quantity* of non-local information with which the agent makes decisions. If the agent only coordinates with a handful of other agents, it probably has a limited view of the overall global picture, however, this says nothing about the quantity of the global view that is necessary to make good decisions, i.e., a very limited view may still provide all the information needed to make rational decisions. We return to the role of structural information later.

[2] Information such as agents being self-interested with respect to one another or altruistic may be used from both a structural perspective and a valuation perspective. On the valuation side, this information might specify how an agent determines the value or utility of coordinating with another agent (e.g., is it getting paid?). On the structural side, it might specify the coordination protocol that is used, i.e., agents that coordinate in a self-interested fashion might use a long negotiation-style dialogue whereas altruistically interacting agents might employ a simple dialog based on the assumption that the other agent will specify its needs exactly rather than attempting to haggle.

[3] Corkill's work [17] that dealt with structuring the computation of distributed problem solvers is somewhat related to this notion of influencing the behavior of the agents via organizational knowledge. In Corkill's research, agents have several *interest areas* under which all of their activities are grouped. Organizational knowledge (or "meta-level" information) biases an agent's behaviors with respect to the interest areas, i.e., it sets parameters used by actions in the particular interest areas. For example, it might partially specify the agents with which a given agent will communicate, or the types of information that the agent sends to other agents. Corkill's research is framed in terms of a distributed problem solving application (homogeneous problem solvers) and does not address valuation issues specifically, but the notion of *modulating* or *biasing* local problem solving via organizational knowledge (leaving flexibility in tact) is of interest to this thesis.

It is important to make the distinction between this proposed work and related interests of the local research community. In [61] Lesser discusses a three tiered architecture in which decision making occurs in three different and semi-autonomous layers that operate concurrently within an agent. The layers are: 1) an organization layer in which decisions of a long temporal scope are made (the formation of alliances between agents), 2) a small-agent group coordination layer where decisions of a shorter scope and duration are made, and 3) a local agent scheduling layer where decisions have a short temporal scope but are very precise. The knowledge structures proposed in this thesis relate to all aspects of Lesser's layered view of agent control. However, the decision process relates mainly to the small-agent and local agent control levels because it is not intended to deal with the high-level reasoning necessary to form (meaningful) organizations. Instead, it is intended to incorporate organizationally driven knowledge, specified by an organizational design component or *a priori* by an organizational designer [4] , in the agent's decision process.

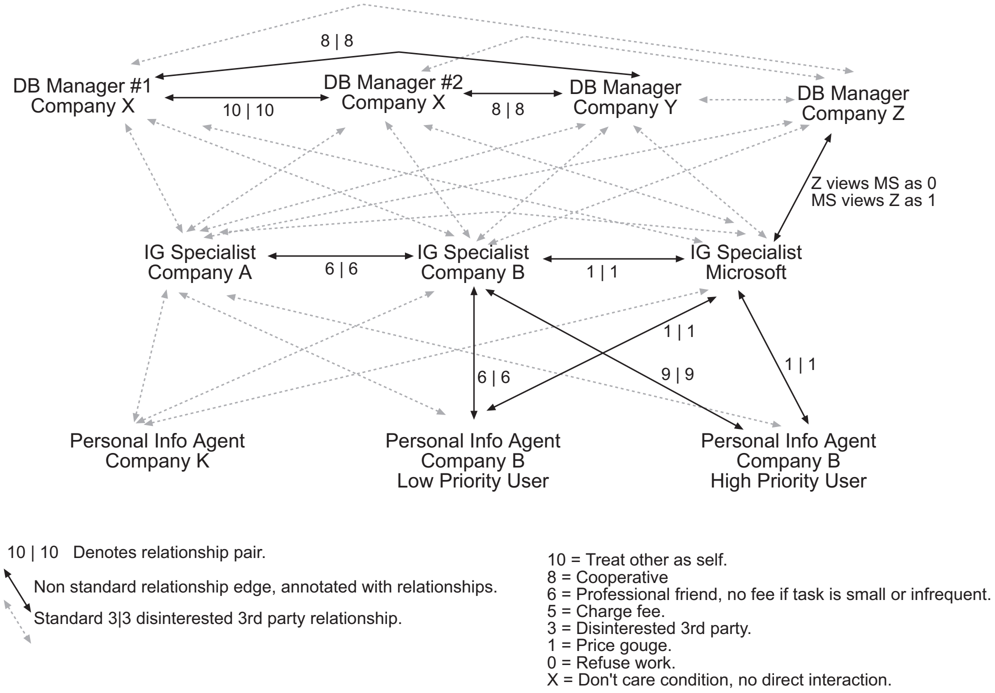

8 | 8
DB Manager #1 Company X
10 | 10
DB Manager #2 Company X
8 | 8
DB Manager Company Y
DB Manager Company Z
Z views MS as 0
MS views Z as 1
IG Specialist Company A
6 | 6
IG Specialist Company B
1 | 1
IG Specialist Microsoft
1 | 1
6 | 6
9 | 9
1 | 1
Personal Info Agent Company K
Personal Info Agent Company B Low Priority User
Personal Info Agent Company B High Priority User
10 | 10 Denotes relationship pair.
Non standard relationship edge, annotated with relationships.
Standard 3|3 disinterested 3rd party relationship.
10 = Treat other as self.
8 = Cooperative
6 = Professional friend, no fee if task is small or infrequent.
5 = Charge fee.
3 = Disinterested 3rd party.
1 = Price gouge.
0 = Refuse work.
X = Don't care condition, no direct interaction.

Figure 2.2: An Organized Network of Interacting Agents

To illustrate the types of problems we would like to address in this thesis, Figure 2.2 shows an organized network of interacting information agents in the WARREN [19] style. The agents are highly specialized and interact in different ways, reflecting their different relationships. Some relationships are influenced by corporate connections, e.g., the

[4] The "organizational design work" done by Garvey, Benyo, and Lesser also does not relate to Lesser's uppermost level of organizational design. Said organizational design work is actually an instance of situation specific conditioning and the generation of default knowledge – it too pertains mostly to the coordination and scheduling levels. I personally feel that the organizational level discussed by Lesser should pertain to forming alliances, and power-relationships, between agents, not simply specifying interactions between tasks or determining which tasks an agent is likely to perform. Said pre-specified nles and task assignments are part of the organizational role discussed in this document, indeed, they fall into the "structural" category above, but the generation of these is simply an analysis of the distributed computation *at the exact same level* as the analysis done by the coordination modules and the scheduler. I feel that the problem is in approaching the issue from within the existing TÆMS framework rather than constructing new structures as proposed here. Using the structures and support provided by this thesis, a true organizational design component could be created that constructs organizational roles, but also constructs organizations of agents, operating on a more abstract level than that afforded by TÆMS and the current "design component."

database manager agents for company X are mutually cooperative and they extend a slightly lesser degree of cooperative behavior to the agents belonging to company Y, a subsidiary of X. In contrast, the database manager for company Z will not service requests from the Microsoft information gathering specialist. A different type of relationship is that between the information gathering specialist for company B and the IG specialist for company A – they have a good professional relationship and will cooperate, doing tasks for free, with one another as long as the tasks are not too large or occur too frequently. We return to this figure again in Chapter 5, the intent is to illustrate the inherent complexity in even a small network of cooperating agents. The ability to quantify the relationships between the agents, and to understand how the relationships relate to the value associated with different possible courses of action for each agent, is central to building larger "real world" multi-agent systems. In this example scenario, how would the IG specialist for company B decide between servicing requests made from the company A specialist, the two company B users, and the Microsoft IG specialist? The IG specialist may not like Microsoft, but, maybe Microsoft is paying a considerable amount for service. How does this financial value compare to the goodwill obtained by servicing the request from the company A specialist to the inter-corporate goodwill obtained by processing the inter-company requests? This thesis will attempt to frame and address issues such as these.

2.2 Distributed Computational Processes

One possible view of the problem solving processes[5] carried out by a set of individual agents is that at any moment in time there is a global structure that represents and relates all of their problem solving activities, including full elucidation of task interactions. If such a global structured existed, and the environment held constant so the structure did not evolve or change, the optimal result for the system as a whole could be computed by selecting the optimal (according to some utility measure) next primitive action for each associated agent and repeating this process until all goals had been achieved. Figure 2.3 illustrates this concept; the tasks of three agents are merged in the construction of a single task structure. Conceptually, this is the objective of coordination.

The problem is that, due to bounded rationality [71], it is impossible for one agent to construct such a view. Even if a complete global view could be constructed, finding the optimal course of action for each agent is computationally infeasible (for any non-trivial structure). Thus, the objective in most agent coordination mechanisms that operate in real domains is to approximate the global view through the exchange of selected information between agents and to perform approximate analysis on the partially observed global structure to determine a course of action. In general terms, the goal is to approximate the global structure and approximate the hypothetical globally optimal action-selection-process through a satisficing local selection process. To complicate matters, the environment generally does not remain constant and in most real applications the goal sets of the associated agents change over time, thus the global view too must change over time. Even if an agent could determine a locally optimal course of action at a given instant, continuing the course of action may not be optimal a moment later.

The main questions that arise when framing the coordination problem in this fashion are: 1) how to bring in enough of the non-local context in order to construct a "good" (having the most important elements) approximation of the global structure, 2) how to approximate the globally optimal action selection process locally, based on the limited view available to the agent, 3) how to balance information gathering actions that form a more complete view of the global structure with the problem solving actions that actually accomplish the goals (akin to deliberation versus action). This thesis addresses these issues by increasing the amount of detail that the agent has about the non-local process and

[5] This view finds its roots in the distributed problem solving (DPS) aspect of multi-agent systems. While the view is valid and a useful reasoning tool, it is important to remember that the global structure is potentially highly dynamic. In many applications change is due not only to changes in the environment, but also due to changes in the goals pursued by individual agents. In our general multi-agent coordination work, we are interested in the interactions of heterogeneous agents in highly dynamic environments. This is in contrast to the distributed problem solving view where the agents are mostly homogeneous and share a common goal (solve the problem) and dynamism is data driven (new data arrives, processing produces new hypothesis). Viewing the process as being primarily data driven might leave out certain prescriptive structures and may be the reason that joint goals are frequently *implicit* in coordination that originates with a DPS view of the problem.

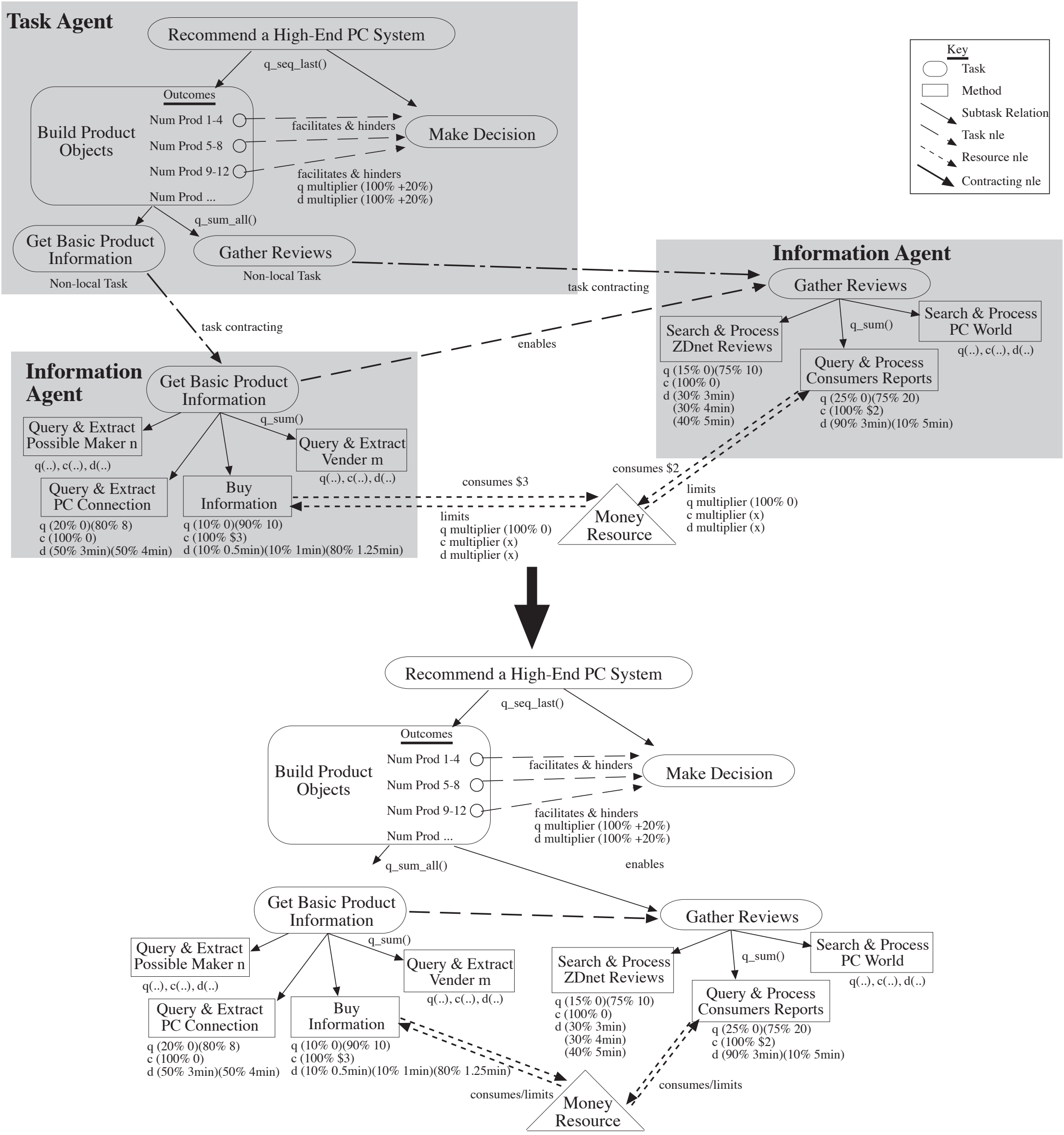


Figure 2.3: Construction of a Global Task Structure from Distributed Structures

by providing richer information from which to measure the value of particular actions, including coordination actions and domain actions. By providing more detail about the way in which the agents relate and are organized, the local agent can reason about the importance of expending energy obtaining particular aspects of the global task structure. For example, if agent α knows that its relationship with agent β only pertains to a single unimportant task (unimportant to both agents), and time is short, α can conserve its computational energy and choose not to construct the part of the global view that includes the unimportant task (thus avoiding the cost of coordinating with β).

It is important to realize that expanding the amount of information used to represent agent activities also expands and changes the hypothetical global task structure. Relatedly, having more detailed views and imposing more structure on the global task structure also potentially changes the types of activities used by the local agents to construct their view of the global structure, i.e., the potential exists for coordination mechanisms that detect organizations and interactions between the organizations (rather than between tasks).

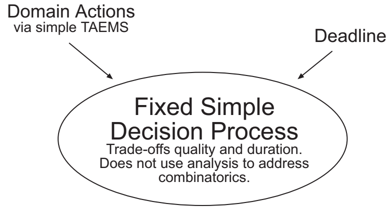


Figure 2.4: Design-to-Time Decision Process

2.3 Decision Processes Revisited

The heart of this thesis effort, indeed of any effort that addresses scaling-up agent control methodologies, is the agent control decision process. The ability to coordinate rationally, and act rationally, is dependent on the ability to analyze the different options available to the agent and to select a set of actions based on this analysis. Make no mistake, this thesis is about building multi-agent systems. But the work will not center on new robust protocols or new brokering conventions, instead this research will concentrate on the most intellectually challenging set of issues facing the community today, the representation of organizational knowledge and its use in the local agent decision process, and, by transitivity, in the coordination process as well. Figure 2.1 illustrates the central nature of the decision process. The local agent decision process drives, motivates, supports, and even implements, coordination between agents. In contrast, coordination mechanisms or protocols deal with the exchange of information necessary to understand how the agents' activities relate. Coordination is meaningless without the ability to reason about the motivation, costs, and benefits of joint work. The converse is also true, without the non-local perspective provided by the coordination mechanisms, the local agent decision process see only the world of the local agent. That being said, the main intellectual challenges reside in the area of knowledge structuring and local decision analysis based on this knowledge. In the future, we will explore the implications of this new wide-context reasoning ability, and the implications of the new organizational knowledge and structure, on the coordination protocols themselves.

This thesis is about expanding the agent decision process to include a larger context – the organizational context in which an agent's computation is situated. We have already begun with the expansion of the knowledge structures used in the Design-to-Time decision process, shown in Figure 2.4, to the current Design-to-Criteria process, shown in Figure 2.5. In this first stage of expansion, the decision process was enhanced to include multi-dimensional goal criteria rather than a hardcoded fixed objective, and to use this criteria to help control the combinatorics of the scheduling problem. The result is a process that is fully targetable depending on the context (e.g., can trade-off in any dimension

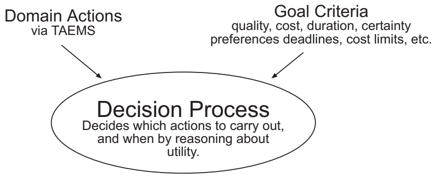

Domain Actions
via TAEMS
Goal Criteria
quality, cost, duration, certainty
preferences deadlines, cost limits, etc.
Decision Process
Decides which actions to carry out, and when by reasoning about utility.

Figure 2.5: Current Decision Process

of the criteria) and has been used effectively in many applications; the new process also executes in soft real time because of the complete integration of the criteria mechanism into the decision process. In this thesis, we will expand on the decision process even more, evolving it to the process pictured in Figure 2.5. This evolution involves three aspects of our current agent control research, the TÆMS modeling framework, the Design-to-Criteria scheduling system, and the GPGP multi-agent coordination module, though each is related differently:

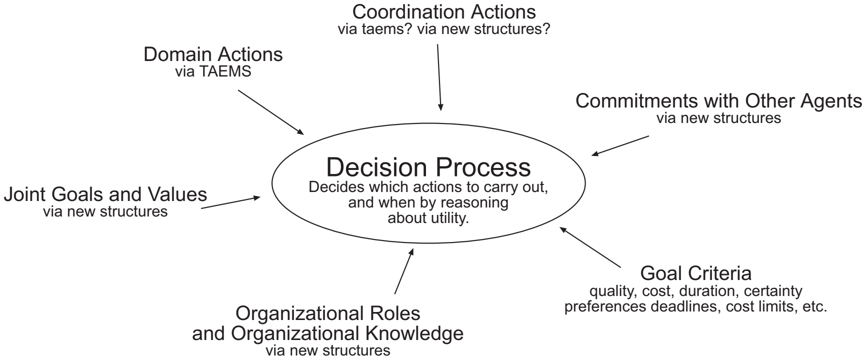
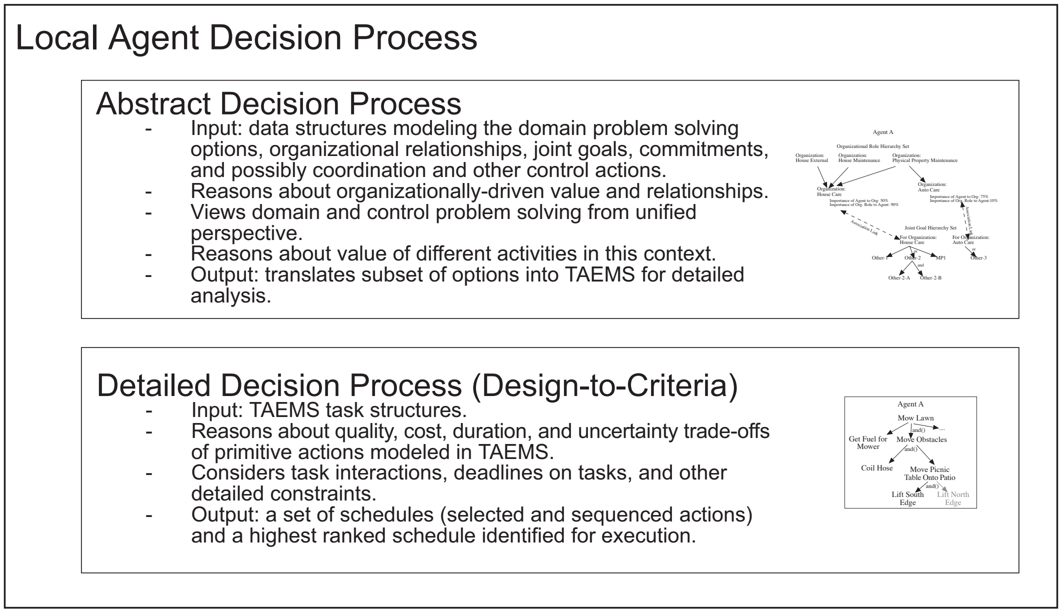

Coordination Actions
via taems? via new structures?
Domain Actions
via TAEMS
Commitments with Other Agents
via new structures
Decision Process
Decides which actions to carry out, and when by reasoning about utility.
Joint Goals and Values
via new structures
Goal Criteria
quality, cost, duration, certainty
preferences deadlines, cost limits, etc.
Organizational Roles
and Organizational Knowledge
via new structures

Figure 2.6: Enhanced Decision Process

TÆMS Task Modeling Presented in greater detail in Chapters 3 and 4, TÆMS is a hierarchical domain-independent modeling language used to represent agent problem solving activities. The most notable features of TÆMS include: 1) the statistical characterization of primitive actions in terms of quality, cost, and duration (via discrete probability distributions), 2) the quantified representation of hard and soft interactions between tasks, and 3) the explicit representation of alternative ways to perform tasks. Historically in our work, TÆMS has been used to model only domain problem solving actions, control actions, such as coordination actions, were not modeled and not considered explicitly in the agent decision process. Modeling control problem solving activities along with domain activities, and reasoning about the trade-offs between each, is desirable, but, also difficult in TÆMS for a variety of reasons including the combinatorial complexity of such reasoning and the difficultly in relating domain derived value to the cost/benefits of control actions.

In this thesis work we will develop a new set of knowledge representation structures, possibly akin to TÆMS models, to represent and relate the different structures (e.g., organizations and their relationships, organizational roles held by agents, joint goals, commitments). The structures themselves may be fairly simple – the overall objective is to create mappings from the individual structures (including TÆMS) into a unified, value or utility based, model from which a decision process can relate value from domain actions to value from coordination actions (or satisfying a commitment) and decide a rational course of action accordingly. The new knowledge will not be integrated into TÆMS (rationale given in Chapters 5 and 8), but, TÆMS may still be used as the representation from which detailed analysis and sequential schedules are produced.



Local Agent Decision Process

Abstract Decision Process

- Input: data structures modeling the domain problem solving options, organizational relationships, joint goals, commitments, and possibly coordination and other control actions.
- Reasons about organizationally-driven value and relationships.
- Views domain and control problem solving from unified perspective.
- Reasons about value of different activities in this context.
- Output: translates subset of options into TAEMS for detailed analysis.

Agent A
Organizational Role Hierarchy Set
Organization: House External
Organization: House Maintenance
Organization: Physical Property Maintenance
Organization: House Care
Importance of Agent to Org: 50%
Importance of Org. Role to Agent: 90%
Organization: Auto Care
Importance of Agent to Org: 75%
Importance of Org. Role to Agent:10%
Association Link
Association Link
Joint Goal Hierarchy Set
For Organization: House Care
For Organization: Auto Care
or
or
Other-1
Other-2
MP1
Other-3
and
Other-2-A
Other-2-B

Detailed Decision Process (Design-to-Criteria)

- Input: TAEMS task structures.
- Reasons about quality, cost, duration, and uncertainty trade-offs of primitive actions modeled in TAEMS.
- Considers task interactions, deadlines on tasks, and other detailed constraints.
- Output: a set of schedules (selected and sequenced actions) and a highest ranked schedule identified for execution.

Figure 2.7: Decision Process Composed of Two Decision Processes Operating at Different Levels of Abstraction

Design-to-Criteria Scheduling Design-to-Criteria *is* the action-selecting-sequencing decision process described earlier. However, as we discuss in Chapters 3 and 4, Design-to-Criteria only entails the analyzes of domain problem solving actions (modeled in the TÆMS language). The incorporation of the quantified, organizationally centered information into the decision process will probably require the creation of a new agent decision process, though the new process may share some common methodological characteristics with Design-to-Criteria (approximation, satisficing, goal directed, etc.). A likely solution path for the new decision process is to view domain actions and control actions from a unified, but more abstract perspective. The new process may still rely on Design-to-Criteria to perform detailed analysis and action selection/sequencing, possibly by translating portions of its unified abstract view of the actions into TÆMS for detailed analysis by the scheduler (or a descendent thereof). If this approach is used, Figure 2.7, it is likely that the two decision making components (that are operating at different levels of abstraction) will interface via a two-way question-and-answer mechanism. It is easy to envision the Design-to-Criteria scheduler discovering during scheduling that it needs more detailed information about other candidate actions, possibly because the candidate actions selected by the abstract view cannot be scheduled due to constraints or interactions not dealt with by the more abstract decision process. Similarly, one can also easily see the benefit of the more abstract organizationally-centered decision process querying the scheduler from time to time for detailed analysis as it determines the value of actions. This is akin to Simon's [71] notion of the organizational structure influencing the objective or utility function.

GPGP and $GPGP^2$ While the relationship of this thesis direction to Design-to-Criteria is fairly straightforward, the relationship of the new knowledge structures and associated decision process to GPGP is more subtle.

Part of this thesis objective is to support new coordination mechanisms and protocols that incorporate and utilize notions of organizational context. In a conceptual sense, the decision process direction of this thesis work relates as much to the original GPGP as it does to the recent Design-to-Criteria scheduling research. Conceptually, GPGP brought non-local information to the attention of the local scheduler (then it was the Design-to-Time scheduler). This non-local information took the form of commitments given by other agents, partial views of the task structures belonging to other agents, and commitments offered by the local agent to other agents. This is appropriate – the coordination module is generally responsible for maintaining relationships with other agents and gathering non-local information. However, the problem with the GPGP/Design-to-Time model is that the non-local information was a second class object to the scheduler and not evaluated in the same light as were the local candidate actions. Thus, when the scheduler evaluated different possible courses of action, it did not reason about commitments given to other agents directly during all aspects of the scheduling process. [6] Instead, the scheduler would produce a set of candidate schedules where the set may, or may not, include schedules that satisfied commitments given to other agents. In a very real sense, Design-to-Time did not reason about the cost/benefit of satisfying commitments given to other agents at all. It simply scheduled local activities and then passed a set of candidate schedules back to the coordination module.

Given the set of candidate schedules, the coordination module (GPGP) would then select one that best met both the local, and non-local, concerns *from its perspective*. In effect, GPGP would second guess the local agent decision process and it would do so without a very good view of local problem solving options. This worked well when the decision process was based on a simple function of maximizing quality within a given deadline, and if we assume that schedule production was exhaustive and thus some schedules that satisfied given commitments were produced. However, as the decision process has become more complex, in order to address real application domains, the act of second guessing the decision process becomes less desirable. With the design of $GPGP^2$ we have come to recognize that the original GPGP/Design-to-Time model does not scale-up to more complex task structures coupled with the more complex goal criteria used in Design-to-Criteria. Because the non-local information is a second-class-citizen to the scheduler, it is possible for the scheduler to produce a set of candidate schedules that do not adequately address GPGP's "hidden" agenda thus leaving the GPGP decision process with few alternatives from which to balance the local and non-local concerns. The recognition of the problems with the GPGP/scheduling coupling, and the partial correction of it with the design of $GPGP^2$, is one of the immediate contributions of this thesis.

To reiterate and clarify, the GPGP / Design-to-Time model suffered from two major problems: 1) Non-local information and commitments given to other agents were not reasoned about directly in the Design-to-Time decision process. The implication of this is that the scheduler was unable to actually decide which schedule to execute because it lacked an appropriate view that included both local and non-local concerns. 2) Because the scheduler did not regard commitments as first class objects, and it could not decide which schedule to actually execute, the coordination module would decide which schedule to execute based on its valuation of the outstanding commitments. The decision process was split between the two modules. On one hand, the decision expert lacks the information and reasoning support to actually deal with the non-local information appropriately, on the other hand the coordination module lacks the ability to *construct* different courses of actions and it even lacks the ability to evaluate different *trade-offs* as are found in the current Design-to-Criteria scheduler. Thus the decision is ultimately being made by a component ill suited to making the decision (GPGP) and the component responsible for reasoning about different possible courses of action does so without truly considering the commitments given to other agents.

In a very real sense, this thesis recognizes the interplay between the local decision process and the non-local information and concerns managed by GPGP. In this new work, non-local information will be a first class object

[6] Specifically, Commitments in Design-to-Time (DTT) are reasoned about during method sequencing, that is the scheduler is weakly biased toward satisfying a commitment when determining in which order to perform a given set of actions. However, DTT does not reason about commitments when determining which actions to perform in the first place (the alternative generation and selection phase), thus it is possible to construct schedules that satisfy no commitments at all.

to the decision process, thus removing the need for the coordination mechanisms or coordination controller to second guess the decision process. One view of the new proposed process is that it is moving the decision functions performed by GPGP[7] out of the coordination module and into a different locus of control that is coupled with the scheduler, in much the same way that the original GPGP was coupled with the Design-to-Time scheduler. The difference is in the clear separation of functionality, and in the types of decisions made and the information that is used in the decision making process. With this new work, the agent decision process is carried out by decision experts, not coordination experts, and the information necessary to make rational decisions is conveyed to the decision process and used internally by the decision process to evaluate different problem solving options.

The relationship between $GPGP^2$ and the thesis is more straightforward. As $GPGP^2$ does not second guess the local decision process, and instead focuses on the issues of coordination protocols and information exchange, $GPGP^2$ is important to this thesis primarily as a demonstration vehicle, i.e., we will integrate the decision processes with the new $GPGP^2$ coordination module as a means to demonstrate the representational power of the reasoning processes in a multi-agent system.

2.4 Role in an Agent

Figures 2.8 and 2.9 illustrates the role of the new decision process component in a typical agent architecture. For the remainder of this section, we will use the term *decision component* to identify the component pictured in Figure 2.7. Though the agent architecture may be significantly more involved, the figure portrays the three major control components and identifies their roles. The agent domain problem solver or planner is responsible for the domain expertise necessary to carry out the selected tasks. For example, in an information gathering application the planner would be an expert in the process of gathering information, i.e., it would know how to search sites, know which sites are likely candidates for a particular type of data and so forth. The domain expert's internal representation of the process or plan is then translated into a TÆMS task structure and communicated to the decision component. In our current agent work, the Design-to-Criteria scheduler is the decision component and it is the scheduler's job to determine a particular (sequential) course of action from the alternative actions represented in the task structure and to return this schedule to the problem solver for execution. The new decision component will continue to fill this role, i.e., it will still perform the trade-off analysis of the domain actions and determine a course of action that best addresses the needs of the agent. However, in the new work, the decision component will also utilize detailed non-local information (provided by the coordination component, e.g., $GPGP^2$) to determine the contextual value of actions. From the perspective of the domain problem solver, the decision component may not change at all.[8]

The interaction between the decision component and the coordination module, in this new system, however may change significantly. In the initial implementation (and for the purposes of developing our decision models) we will assume the organizational information is available at the local agent, e.g., it knows its roles, the relative importance of the different organizations to which it belongs, its relationships with other agents and show forth. However, this is only a first step. To truly create scalable MAS, the agent must be able to communicate with other agents, form organizations on the fly, and even discover organizational relationships with other agents. In the former case, the organizational knowledge is stored in local data base. However, in the later case, the knowledge must be (at least partly) generated by the coordination module through interaction with other agents. Regardless of where the organizational information

[7] Relative to the Design-to-Criteria scheduler, GPGP's decision process was very simple – the discussion here is intellectual and conceptual.

[8] There is a possibility of a richer two-way interaction between the decision component and the domain problem solver. It is possible in certain situations that the domain problem solver will abstract the domain process for the decision component, thus not giving it a full view of the possible options (this is deliberate – it is to help in controlling the combinatorics and focusing the decision process on likely solution paths). In these cases, after evaluating the constraints (both local and non-local) it is possible that the decision component will need to request more detail from the problem solver in order to expand its viable options or to better refine its valuation of the different options.

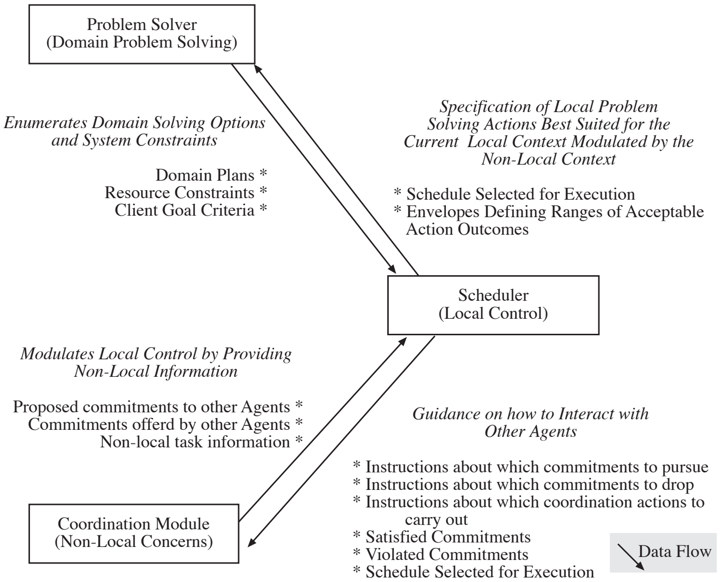

Problem Solver
(Domain Problem Solving)
Enumerates Domain Solving Options
and System Constraints
Domain Plans *
Resource Constraints *
Client Goal Criteria *
Specification of Local Problem
Solving Actions Best Suited for the
Current Local Context Modulated by the
Non-Local Context
* Schedule Selected for Execution
* Envelopes Defining Ranges of Acceptable
Action Outcomes
Scheduler
(Local Control)
Modulates Local Control by Providing
Non-Local Information
Proposed commitments to other Agents *
Commitments offerd by other Agents *
Non-local task information *
Guidance on how to Interact with
Other Agents
* Instructions about which commitments to pursue
* Instructions about which commitments to drop
* Instructions about which coordination actions to
carry out
* Satisfied Commitments
* Violated Commitments
* Schedule Selected for Execution
Coordination Module
(Non-Local Concerns)
Data Flow

Figure 2.8: The Role of the Decision Process Component in an Agent

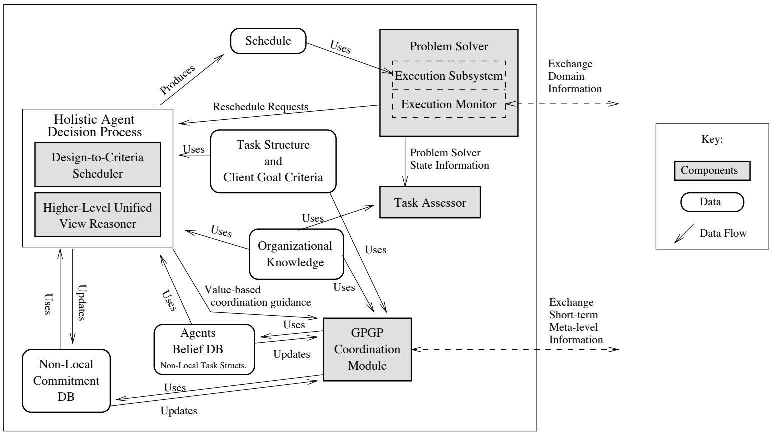


Figure 2.9: Primary Agent Components and Their Interactions

comes from, the interaction between the decision component and the coordination module must change because part of the function of the decision component is to determine over which interactions the module should coordinate (which commitments to form) and possibly even to determine which agents with which to exchange local views. In the new work, the coordination module is an expert at coordination over *targeted* interactions and an expert at constructing the non-local view (as directed by the decision component). Obviously, this is somewhat of an oversimplification, there is a need at some point to exchange non-local information with some agents to *seed* the problem solving process, thus, the decision component may direct the coordination module in a blanket fashion to obtain non-local information from an agent, a set of agents, or even from any available agent.[9] In terms of the decision component's interface with each component:

Interface with Coordination Module The interface between the coordination module and the decision module is as follows:

Input from coordination module to decision module: proposed commitments, non-local task information, non-local commitments. Future input from coordination module (or organization coordination module): new organizational roles or structures.

Output from decision module to coordination module: specifications of which commitments are preferred and which should be avoided, requests for non-local information to gather, commitments that are firm or previously firm commitments that are broken.

Domain Problem Solver The interface between the domain problem solver and the decision module is as follows:

Input from problem solver to decision module: TÆMS task structures, agent goal criteria or problem solving constraints (e.g., deadlines on performance), intermediate results and failure reports.

Output from decision module to problem solver: intended actions or schedules, possibly annotated with intermediate reporting requests and rescheduling triggers.

2.5 Relationship to Joint Intentions, Interaction Theory, and BDI

Much of the formal work in the community centers around the issue of what it means for two agents to have a joint goal and the implications of the goal to agent behavior. This research falls under many names, e.g., SharedPlans [40], joint intentions [14, 11, 7], and joint goals [50, 78] to name a few. In general, this work views agents as autonomous planning entities with interactions between their plans (or intended actions). Agents in this context generally reason *logically* about the existence of interactions and generally adhere to certain behavioral axioms or behavioral constraints with respect to interactions. For example, say two agents α and β each have individual goals to perform task γ, though neither agent can perform the task alone. After exchanging information about their goal structure, the agents then determine that they *might* have a goal in common. The criteria varies from work to work, but generally if there is a dependence relationship between goals shared by different agents, a joint goal is said to exist. The agents generally note the existence of the joint goal in a logical sense in their local knowledge database (or belief space). The goal then comes into play when the agent decides which activities to perform and how it should interact with the other agents under the joint goal. For instance, if α has a joint goal to achieve γ, it cannot elect to perform some other task θ that will preclude doing γ down the road. Similarly, if it performs γ, it must inform the other agent of the performance unless the other agent can observe the results directly.

[9] We need to examine the issue of the asynchronous layered model described by Lesser [61] and its relationship to the increased role of the decision process. While there was little need for the DTT scheduler to be persistent, or the DTC scheduler for that matter, this larger decision component may be a different case.

In some sense, the work on joint intentions and joint goals is closely related to our work. However, it differs in that the focus is on formally defining what it means to have a joint goal, and formally defining the implications of that goal on agent behavior, all from a very coarse-grained general perspective. The joint goal work does not address how an agent chooses from multiple candidate goals, or how the agent decides which actions to perform at a given time, instead it only focuses on the existence of the joint goals and recognition of the joint problem solving activity. To make the point clear, consider an extreme example. Agents α and β have a joint goal to perform γ, however, the joint goal says nothing about *when* either agent will begin work on γ or even that one agent will perform its share of the work in close temporal proximity to the performance of the other agent's task. The joint goal also says nothing about the importance of the γ-centered goal relative to the importance of other goals, individual or joint, that the agents might have. In contrast, our work focuses on deciding *which* goals (or tasks) to perform, *when* to perform them, and *how* to sequence and coordinate the joint activities with other agents – all from a quantitative worth-oriented perspective (rather than a logical one).

The two technologies are actually complementary if structured in a two-leveled fashion. The logical work resides on the upper level, providing reasoning from a coarse-grained or high-level perspective; it gives agents the ability to detect joint goals (from a rigorous formal perspective) and to determine when said goals can no longer be achieved, or are achieved elsewhere, or are no longer even relevant. Our work resides on the lower level and provides the ability to reason about the relative value associated with the candidate goals (detected in the top-level) and the ability to reason about the value of actions associated with the goals, to choose from alternatives, to work on the goals concurrently by interleaving actions, and so forth.

This integrated view may also relate to the question of relating process languages, e.g., [52] to our agent coordination work. It is possible that the process view provides the ability to identify interactions while the decision process and coordination tools provide the ability to reason about the context and to structure the activities of the individual agents.

In this thesis we will not address this possible integration route. However, we will continue to consider this as an avenue of future exploration and continue to evaluate the relationship between joint goal research (and other higher level techniques) and coordination research.

2.6 Research Contributions

We have discussed the ideas of this thesis from a high-level perspective and via their relationship to our existing work. To regroup and summarize, the main goals of this thesis are to:

1. Expand the contextual knowledge used in multi-agent systems to include explicit notions of the organizations in which the agent is situated. The information will include concepts like membership in organizations and the role of an agent in an organization. The information will also specify attributes like the interaction styles used in an organization, the protocols used, etc. In addition, organizational information will have a quantitative aspect that specifies the importance of the agent to an organization, the power relationships between agents within an organization, and the relative importance of particular tasks or actions to the organization.

2. Construct new knowledge structures (in addition to those pertaining to organizations) to explicitly represent joint goals and commitments from a quantified perspective. Modeling and reasoning about the costs of decommitment is also an objective.

3. Incorporate the new contextual knowledge into an agent's local decision process, enabling agents to reason about the relative costs and benefits of particular domain actions, as well as the costs and benefits of coordination-related actions, from a unified perspective. This will enable agents to reason about the interplay between different actions and to evaluate their choices in light of a larger context. For example, an agent will be able to decide

whether the computational effort required to bundle a result and ship it off to another agent is *worth* the value received, particularly when the time could possibly be better spent by performing a different activity.

4. Provide support for future efforts at organizationally-centered coordination. By this we mean the construction of coordination protocols that use the organizational information to modulate coordination activities. For example, to coordinate over a particular joint goal, and agent could examine the organizational role related to the goal and ascertain that agents α and β have similar goals but that γ does not have a similar goal (or even an interacting one) and thus may be omitted from the coordination dialogue. As mentioned previously, this is an interesting area of work, but we believe that meaningful incorporation of organizational information must first occur at the decision process level. (Otherwise the agent may know with whom to coordinate, but not appropriately decide which goals over which to coordinate in the first place.)

In terms of our current research, and the research in the field as a whole, the new ground proposed in this thesis makes several contributions. Related work is discussed in detail in Chapter 6. In this section we identify the contributions and place them from a high-level perspective. The main contributions of this thesis to the multi-agent systems community are:

- *Explicit representation of complex organizational structure and information.* Notions of organizations of computational agents date back to work in distributed problem solving, however, the organizational relationships and associated organizational roles presented in this thesis are more complex structurally and more detailed than any knowledge structure studied by the field to date. Even very recent work in obligations (Section 6.1.5) does not incorporate notions of different coordination approaches for different groups of agents or the association of organizational relationships with artifacts related to them like joint goals and commitments. A notable example of related work to date is Durfee's behavior spaces [32] (discussed in detail in Section 6.1.4) in which agents activities are abstracted in to a knowledge structured that identifies who, what, where, when, how and why.

- *Quantified view of agent relationships.* Recent research that addresses more complex agent interactions and relationships, e.g., rights [65] (Section 6.1.2) and obligations (Section 6.1.5), focus on these constructs as being binary or logical. The quantified view of agent relationships enables us to reason about the contextual importance or value stemming from the relationships. For example, a task assigned by one's superior at work is more important than running an errand for a friend, but if the same friend needs an emergency ride to the hospital, the task for one's boss is suddenly less important.

- *Quantification of commitments, decommitment, and joint goals.* Akin to the above, work in the field seldom deals with notions like a degree of commitment or the relative importance of a particular joint goal (particularly work in joint intentions). Research typically defines what it means to have a joint goal, and the implications to the agent (e.g., an agent cannot arbitrarily decide not to work on a joint goal). However, little work deals with these constructs from a quantified perspective or discusses the context in which it might not be *worth* satisfying a commitment or working on a particular joint goal. With respect to decommitment, it is important to realize that decommitment penalties, as with commitment value, must also be determined by the larger context, e.g., the values of the tasks assigned to the agent, the importance of its organizational roles, other candidate commitments being considered, other commitments that have been formed, and so forth. Attaching a penalty to decommitment as well as associating value with commitment is important because it gives the overall problem solving process a measure of stability. Without a penalty for decommitment, agents may form commitments with impunity and break them when better opportunities are presented. This impacts the entire problem solving network as agents must plan, temporally, based on assumptions about when results and other needed items will be provided by other agents. Each time a commitment is broken, it potentially impacts every other commitment that has been formed since the time at which the newly broken commitment was formed – it can even affect the agent breaking the commitment through chains of commitment/task relationships.

- *Relation of quantifications stemming from agent relationships and organizational levels to the value and importance associated with joint goals, commitments, and primitive actions associated with the relationships.* In this thesis we take the position that there is a quantifiable relationship between high-level agent relationships, joint goals associated with these relationships, commitments associated with the joint goals, and primitive actions associated with the commitment. Furthermore, we take the position that the value of these items stems from three different directions, 1) its individual value (e.g., a primitive action has some initial value or properties of its own), 2) the influences of other items upon it, e.g., a very important organizational role might elevate the importance of goals associated with it, and thus the importance of commitments and primitive actions that serve to satisfy the goals, 3) the current problem solving context, including environmental factors. We believe this quantified relationship network is unique in the field.

- *Incorporation of the quantified organizational characteristics into a contextually sensitive local agent decision process.* In all but the most recent research, agents make decisions about what to do, with whom, and when, using knowledge of their own tasks and goals and knowledge of the other agents' tasks and goals. This approach is fine for small groups of agents, but, it does not scale up well to lager organizations and structures of agents where the agents may interact in different ways with different agents. For example, an agent may coordinate for financial gains with one non-group agent but for altruistic reasons with a group agent. In this thesis, we incorporate organizational knowledge into the agent reasoning structure and design (and implement) analysis tools that use the structure to help the agents make value or utility driven decisions about which actions to perform and when to coordinate. Additionally, we approach the problem from a dynamic perspective. As the problem solving context changes, e.g., new relationships are discovered, resources become more widely available, etc., so too does the appropriateness of a particular chosen course of action. The contextual evaluation of the utility of actions, joint goals, commitments, and even agent relationships, is a major contribution of this work. As with our work in Design-to-Criteria scheduling, the ability to re-evaluate decisions and to redefine what is desirable based on the current context, when the entire problem solving situation changes (not just the bindings, i.e., this is not a Bayes network with changing influences), is unique.

- *Use of a domain independent perspective.* Our work is also notable in its attempt to tackle these issues from a domain independent perspective. The continued success of using TÆMS and Design-to-Criteria in different domains is a testament to the notion that there are general approaches to control problem solving that are amenable to multiple applications. (Though we would never argue that situation specificity is a detriment to problem solving either.) The model-based domain independence is also important from an infrastructure perspective. In order to lower the development cost of MAS, off the shelf control problem solvers and tools like Design-to-Criteria are invaluable.

- *Support for organizational coordination protocols.* While this topic is beyond the scope of this thesis, it is a natural and planned extension to this work. Little work has been done in the development of coordination protocols for the formation of coalitions or organizations of agents, and most of the existing work is not grounded in actual MAS where the end result is the detailed coordination of agents. The work in this thesis builds the foundation on which such grounded organization formation protocols can be explored.

- *Support for real-time coordination.* By reasoning about the value of coordination activities and communication actions (possibly), we set the stage for time dependent trade-off analysis of these actions. This would enable agents to reason about their coordination activities from a temporal perspective (based on predictions of the outcomes of such actions) and to target their activities to address particular time or resource constraints. It is possible that if a two stage decision process is constructed that entails describing coordination and communication actions in TÆMS for the Design-to-Criteria scheduler that one outcome of this thesis is the ability to address resource constraints within that limited context.

This thesis and work stemming from or motivated by it also make contributions to the local research agenda. These are:

- *Treatment of commitment as a first class object.* Currently agents do not value commitments made with other agents in the same way that they do their other problem solving options. Commitment, and the value obtained from commitment satisfaction, is an extra or add-on to the agent's reasoning process. In this thesis, we will integrate and extend notions of commitment so that commitments, local activities, coordination activities, etc., are all reasoned about using a uniform, value driven framework.

- *Separation of the coordination analysis process, and its supporting knowledge structure, from the protocol used to carry out coordination-centric communication between agents.* Historically, in our work and in the work of many of our peers, certain aspects of the analysis process required to determine when to coordination, with whom, and to what extent, has been embedded in a body of code called a *coordination mechanism*. Our recent work, $GPGP^2$, has taken steps to separate the coordination protocol from the supporting analysis code and reasoning frameworks so that protocols can be extended easily, replace, specialized, etc. This separation also helps us to better understand – more clearly view – the analysis aspects of the coordination problem.

- *Explicit recognition and representation of joint problem solving activities.* In previous work, the notion of a joint activity was somewhat implicit or embedded in the knowledge structure (TÆMS) used to model agent activities. This implicit view of joint activity makes coordination difficult in particular cases. In this thesis we will extract and expand on the concepts of joint activity and explicitly represent, track, and reason about joint goals (goals that agents have in common). This will enable us to leverage a large body of formal and theoretical work on the subject, as well as providing a clear and principled basis from which to motivate coordination between agents.

- *A step toward support for behavioral axioms.* Much of the formal work in the area of intention and commitment (aka responsible action) [12, 7, 40], seems to translate, at the practical level, into behavioral axioms [50, 81] that define the way in which socially responsible agents behave. Part of this thesis may entail understanding the mapping of formalism to practice and adding axiomatic support to the agents knowledge and control structures, at least at the organizational level.

In this chapter we introduced the thesis topic and discussed its relationship to existing work in the field from a high-level perspective. In the next chapter, we provide a background of multi-agent systems and identify key challenges faced by the field. In Chapter 4 we describe the current state of our work and identify its relationship to the topic. Chapter 5 presents the topic in more detail and Chapter 6 discusses highly relevant related work. Chapter 7 presents work and experimentation plans.

Chapter 3

Introduction to Multi-Agent Systems

Chapter 2 introduces the thesis topic and describes the intellectual issues and contributions of this proposed research. This chapter is intended to familiarize committee members with the subdiscipline of multi-agent systems and software agents, and to relate large bodies of multi-agent research to the thesis topic. Subsequent chapters provide more detail about our local multi-agent work, Chapter 4, the topic, Chapter 5, and research in the field, Chapter 6, that strongly relates to this research agenda. A work plan and experimental plan are presented in Chapter 7.

3.1 Introduction to Software Agents and Multi-Agent Systems

This thesis deals with multi-agent coordination, which is, literally, about getting different software agents to work together in a coherent fashion. Let us postpone the issue of what we mean by "software agent" for the moment and consider an example involving a couple of humans, John and Al. Say John needs to go to the Amherst Post Office to mail a package and Al needs to pick up a prescription at CVS (CVS is right near the post office). John and Al can each perform their separate tasks, however, if they communicate their planned actions and reason about them, they will realize that Al can perform John's task with a slight extra cost to Al, but a huge savings for John. The process of communicating their intended tasks, recognizing the partial overlap in their objectives, and determining who is going to do what, and when, is multi-agent coordination. In this case the agents are simply humans rather than software agents.

Consider another example. This time John is moving to a new home and Al is helping. John, being quite musical, has an upright piano and it too needs to be moved. John and Al discuss the piano (while Al leans on it and wipes the sweat from his brow) and they decide that they will each grab an end, carrying it by its sides rather than a front/back arrangement. John and Al then lift their ends concurrently and carry the piano out the door to the van. The coordination episode in this case is slightly different from the episode above – this time the two had to act in concert, rather than one performing a task for the other. In our work we deal with both of these types of coordination (and others) using the notion of a *commitment*. In the first example, Al gave John a commitment to mail his package and John trusted that Al would faithfully perform the task. In the second example, John and Al gave (and received) mutual commitments to lift the piano at the same time and to carry it together.

Let us consider a variation on the moving example. Say that Al agrees to help John move, but, not being quite so altruistic as in the previous scenario, Al wants something in return. In this case Al and John may negotiate or haggle over the price of Al's services. Perhaps John agrees to pay Al for his time in money, goods, or perhaps by an agreement that John will do something for Al in the future. In this scenario, Al is behaving in a *self-interested* fashion rather than the *fully cooperative* (altruistic) fashion of the previous example. Self-interest, and degrees thereof, are also issues in multi-agents systems and something that this thesis will address.

How is it possible that we are discussing multi-agent systems without yet defining the term agent? It is because notions of agency have strong parallels with simply being human, however, we can be more specific than that. In the examples above, the humans had choices about what tasks they were going to perform and about when said tasks were to be carried out; the humans were also able to reason about their choices, possibly selecting one from a set of candidate tasks. These two notions are much of what constitutes an "agent." Definitions [49, 54, 10, 78, 61, 48, 30, 89] differ, particularly between the research community and popular computing, but, in general, we will use the term agent to denote software that has most of the following properties:

Multiple Goals or Tasks Agents generally have more than one thing that they may do at a given time, and different ways to go about solving problems. The term *goal* is from an area of artificial intelligence (AI) known as planning – in a general sense, goal simply means "something to be achieved" or "something to perform." This characterization differentiates agents, which are complex problem solvers, from simple software systems that perform tasks like forward email or retrieve news stories.

Choice Given that agents have multiple things to do at a given time, and generally multiple ways to go about doing them, the issue then becomes knowing what to do when. Software agents have a *choice* about what actions to carry out and they make this choice through a *decision process*. Part of this thesis will contribute to the way in which we view an agent's decision process and the types of information that is incorporated into the process.

Autonomy Agents make their decisions (or choices) without the guidance of humans. External factors may *influence* or *condition* the decision process, but, independence and autonomy is an important facet of agency. In a sense, autonomy is as much a requisite for a sophisticated agent as it is a consequence of having a sophisticated agent. The only way to leverage expertise or ability in a particular area is to let the specialist perform the tasks in which they are specialized. Of course, reasonable results may also be achieved if the specialist is directed by another specialist, but then we encounter a problem known as *bounded rationality*. (Next bullet).

Rationally Bounded Agents have a finite amount of processing power, finite amount of storage space, a finite amount of time in which to perform tasks and a finite amount of knowledge about the world or expertise [72]. This particular item is often not obvious to people outside of research level computing. Even though machines continue to get faster, have more storage, etc., most of the tasks that require sophisticated problem solving agents are beyond direct solutions means. They are *computationally infeasible* generally having an exponential number of possible solutions or requiring expertise beyond our ability to understand and encode in the software. Sophisticated problem solving agents generally rely on heuristics, approximations, approximate problem models, etc., to control the combinatorics and to make problems manageable. Thus, not only is it generally not possible for one agent to perform all of its tasks, and those of the other agents, it is also not generally possible for one agent to incorporate all of another agent's sophisticated problem solving code, knowledge, etc., into itself. Unless the agents are homogeneous, all being instances of a given prototype, agents can't generally become other agents. Additionally, because of storage and bandwidth limitations between agents, the transfer of such expertise is a non-trivial process and it may be infeasible.

Explicit Goals Agents explicitly represent their possible activities and reason about them. This is in contrast to an *implicit* goal representation where goals are understood and reasoned about by the programmer *a priori* and simply embedded in the computer code. A PC word processor is an example of a software system that lacks any notion of an explicit goal. Word processors, like Microsoft Word, do not reason about meeting goals or select between multiple possible goals. When a user "clicks" a menu, the menu pops up and the relationship of *click* followed by *pop up* is hardwired into the code. Word does not make a decision to pop up the menu and is not given a choice. (Word is not considered a software agent either, by the way..)

Explicit goal representations are important in sophisticated agents because, as mentioned above, agents generally have multiple different things to do, they are responsible for choosing what to do, and they make this determination through a decision process. The decision process involves reasoning about the (explicitly represented) goals or candidate actions and choosing those that best fit some measure of utility or goodness.

Situated Agents reside in an environment that they can perceive and act upon. In robotic terminology, we call these *sensors* and *effectors*.[1] An example of an effector in a web information-finding agent is the ability to interact with fill in search forms (e.g., at AltaVista, Lycos, or Infoseek); an example of a sensor in such an environment is the ability to measure the network load and timings between different sites.

Persistence Software agents generally persist over time. Upon initialization, they conceptually enter a loop in which they sense the environment, decide what to do, attempt to affect the environment, and then repeat. This does not mean software agents are assumed to have an infinite existence, but, it means that they do not simply carry out some task (e.g., print a document) and then cease to exist. Persistence also does not imply a static locale – in some research [90, 8, 67] agents are mobile entities that move from machine to machine.

Learning Not all software agents learn over time, however, learning goes hand in hand with persistence and with explicit representations of activities. In other words, since they exist for longer periods of time, the ability to adapt to the changing computing environment (e.g., different network performance characteristics) is important.

These concepts can be abstracted into a few key ideas. Agents are 1) *situated* in an environment that they can sense and effect, 2) *flexible*, having choices and responding to the environment, possibly even being pro-active with respect to the environment, and 3) *autonomous*, making choices independently.

Though research in software agents finds its roots in AI in the 70s, it is only recently that agent technology is being used in real-world systems and commercial products. One example of an agent predecessor is the area of expert systems; expert systems are software programs that are disembodied experts on a very specific subject. They are often used to diagnose problems – for example, many of the commercial software vendors use expert systems to support the customer service operators who field calls from users and help to find problems. Expert systems, however, are not considered agents because they are not situated in an environment – they cannot sense the environment directly and do not act upon it. Nor are they autonomous. A human simply describes a problem to the expert system and the system performs its reasoning and emits a response, answer, or diagnosis.

Examples of deployed agent systems include information gathering softbots [30, 60] that go out on the internet and find information for human clients, or perhaps comparison shop to find the best price for their users. Other, more sophisticated examples, include meeting scheduling systems [43], hospital scheduling systems [21], portfolio management systems [20], and so forth [51]. Agent technology is still in its infancy, but, we have reached a point where agent-oriented design and agent-based applications are becoming widespread.

Interestingly, the same advances in popular computing, namely distributed computing and the web, that are stimulating the growth of software agents are also driving expanded interest in multi-agent systems [39, 61, 49]. Multi-agent systems consist of groups of loosely coupled agents that work together on tasks. For example, a group of physically distributed experts collaborating on the design of a new aircraft wing [28] is a multi-agent system. The loose coupling and coarse grain size of multi-agent systems distinguishes them from more traditional computing systems like distributed databases (databases that operate on different machines but are accessed through a single interface). In multi-agent systems, the agents have choices about with whom to collaborate, how to negotiate, what to charge for services, etc. The multi-agent paradigm is becoming increasingly important to the web and similarly constructed information networks where different resources (databases, programs, etc.) are distributed and under the control of different

[1] Some researchers regard software agents as software robots [35, 30], often calling them "softbots." However, this class of research often focuses on more simple agents that lack some of the characterizations above, e.g., autonomy.

corporate entities. One can easily imagine the IBM database server agent negotiating with a Microsoft information finding agent over the price at which information will be supplied!

Research in multi-agent systems covers a broad range of topics. Some researchers focus on economic theories [2, 8, 69, 34] that can be used to set prices for agent-centric goods and services, i.e., to guide agents' behaviors in a self-interested interaction setting. Other researchers focus on the message format or conversation protocol [4, 57, 58, 56] used by the agents to discuss and describe items so they can interact. Obviously, if agents are unable to communicate, they cannot coordinate their activities, bid on desired products, etc. Still other researchers focus on formal notions of intent [14, 50, 11, 7, 40], ontologies for communication [42, 41, 73], and architectures and design paradigms for building agents [78, 92, 10, 61, 70, 91]. These items are all important aspects of building multi-agents systems and many of them provide broad theoretical and practical underpinnings, however, at the ground level, in order for agents to coordinate their activities and interact cooperatively in a meaningful fashion, they must be able to reason about their actions and temporally sequence them. Relating this back to the example in the previous section, John and Al may spend a great deal of energy negotiating Al's price for helping John, and the language in which they converse is certainly important, as is the ability to understand each other, but, when it is all said and done, if John and Al lack the ability to decide which actions to perform, and then to temporally sequence said activities (i.e., lifting the piano), the work can never be accomplished.

We use the term *multi-agent coordination* to describe the process of determining which actions need to be sequenced across agents and then doing the operations necessary to sequence said actions. Most of the work in multi-agent coordination focuses on managing task interactions [24, 22, 76, 77, 80, 93, 50, 32, 31, 40, 21, 82, 36, 37, 33, 38], and the rationale behind this focus is sound – task interaction is the motivation for coordination between agents [62, 27]. Without some form of interaction between the tasks belonging to different agents, there is no need for the agents to work together at all. Interactions can be direct, like the case of Al and John moving the piano, or they can be indirect, through a shared resource like money or network bandwidth. Except in a pure task allocation centered framework, and it is unclear how hierarchical decomposition of tasks could take place in such a framework [2], multi-agent coordination is required. Before we can discuss coordination further, a few points need to be made.

Local and non-local Each agent has a view of itself and this view-of-self is called a *local* view. Relating back to the previous remarks on the characteristics of agents, an agent's view of its own goals is a local view. Agents may also have views of non-local information. This is information about other agents with which the *local* agent interacts. To clarify, if Al is an agent, what Al knows about Al is a local view. In contrast, what Al knows about John is a non-local view. The use of the word "view" here is somewhat artificial. The important item to understand is that *local* denotes beliefs, desires, intentions, etc., that an agent has about itself. *Non-local* denotes beliefs that an agent has about other agents.

Imperfect Information Because of limited bandwidth and bounded rationality, agents generally do not have perfect views of other agents. In fact, they may only have approximate information about the characteristics of their own actions, i.e., estimates of what will happen if they do a certain task. (This is consistent with how humans operate, e.g., one never knows for certain how long one will wait in line at the Post Office.)

Tasks and Actions Hereto we have danced around the issue of terminology in order to avoid burying the reader with definitions, however, definitions are necessary in order to be more precise. *Primitive actions*, or simply *actions*, are operations that the agent performs to bring about some change in the environment (effectors, using the robotic language). For example, an internet agent may query a web site or interact with a fill-in form at a popular search

[2] Consider task contracting based on a fee or bidding system. On the surface, it appears that in a contracting situation agents would not need to temporally sequence their activities, however, if a task is broken down and parts are contracted-out, the assimilation process must happen after the results are produced. In other words, the act of decomposing a task introduces temporal constraints between the tasks that do the work and the tasks that assimilate the results. Temporal sequencing constraints are likely to occur in any hierarchical task decomposition – only where the contracting is for a set of wholly independent, non decomposable tasks, is the temporal sequencing of actions not an issue.

engine. Agents also have *tasks* which, strangely enough, are not directly executable. Tasks are hierarchically decomposed into actions that can be performed. A web agent might have a task to "obtain information about word processors for Windows 98," where the performance of the task involves primitive actions like querying web sites or interacting with fill-in forms (the actions mentioned above). Tasks are generally regarded as an artifact that is created to fulfill a *goal*, or objective, that the agent may have. For example, our web agent might have a goal to "find a word processor for the lab PC" and the goal motivates the task of searching for word processor product information, which is in turn performed by executing a series of primitive actions.

Since the purpose of this introduction is to provide the necessary background for a detailed discussion of the intellectual ideas of this dissertation, let us now discuss our work in coordination from a high-level perspective. In subsequent sections, when the details are out in the open, we will differentiate our work from others in our field. In our recent coordination work in Generalized-Partial-Global-Planning (GPGP) and its extensions [25, 27, 29, 22, 88, 61, 59], and in our contracting [94] and resource-coordination [6] work, the multi-agent coordination process consists of several different types of operations, presented on a flow basis in Figure 3.1. (Note: there are strong parallels between our view on this process and the others in our field, i.e., this is still somewhat general.)

Detection Agents exchange information about the activities that they are planning to perform and information about candidate activities that may be performed sometime in the future. This enables the local agent to determine which of its local tasks interact with the non-local tasks belonging to other agents. The exchanged information describes the tasks statistically, via discrete probability distributions, in three dimensions (quality, cost, and duration). Upon identification of the interactions between the local agent's tasks and the non-local tasks that it knows about, the interactions are also characterized statistically in terms of the effects on quality, cost and duration of the involved actions. We will revisit the details of this later (Section 3.2), the important point is that in our work interactions and tasks are viewed from a quantitative perspective as well as a qualitative one.

Analysis and decision making Not all tasks are equally valuable to the local agent. Recall that agents have choices about which operations to perform and how to perform them. Agents employ an analysis process in which they reason about the implications of performing certain actions and the implications of different orderings of the selected actions. Agents also analyze the value of coordinating, or not, over particular interactions. In some cases it may be critical for one agent to coordinate with another and in other cases it may not be worth an agent's while to coordinate. For example, if Al offers to help John with the piano, but is only available for a very restricted set of candidate times, John may decide to hire movers instead of working with Al because of the amount of work required to find a time slot during which John, Al, and the truck can all come together. The analysis computation has several different facets and it is important in most operations of the coordination process, e.g., when commitments are proposed, when new commitments are formed, when an agent's set of candidate actions change, when a failure occurs, etc. Decision making is coupled with the analysis – through the analysis process the local agent understands the *utility* of different options and it selects a course of action, i.e., decides what to do, based on this utility measure. Generally agents try to maximize utility where the utility is computed by matching the quality, cost, duration, and uncertainty characteristics of a particular course of action against a set of goal criteria [83, 86] or preferences that describe the requirements of the current situation. For example, if the agent is under time pressure to produce a result, the goal criteria would emphasize that low-duration is more important than getting a high quality result. The agent would then reason about its options from this perspective and decide on a course of action accordingly. As we will discuss, the analysis process, embodied in our Design-to-Criteria scheduling work [87, 83, 86, 85, 84], is an area in which our research distinguishes itself from that of the community at large. We take a very detailed quantitative view of analysis and decision making that is generally absent from the other work in our area. This is also one of the main thrusts of this dissertation.

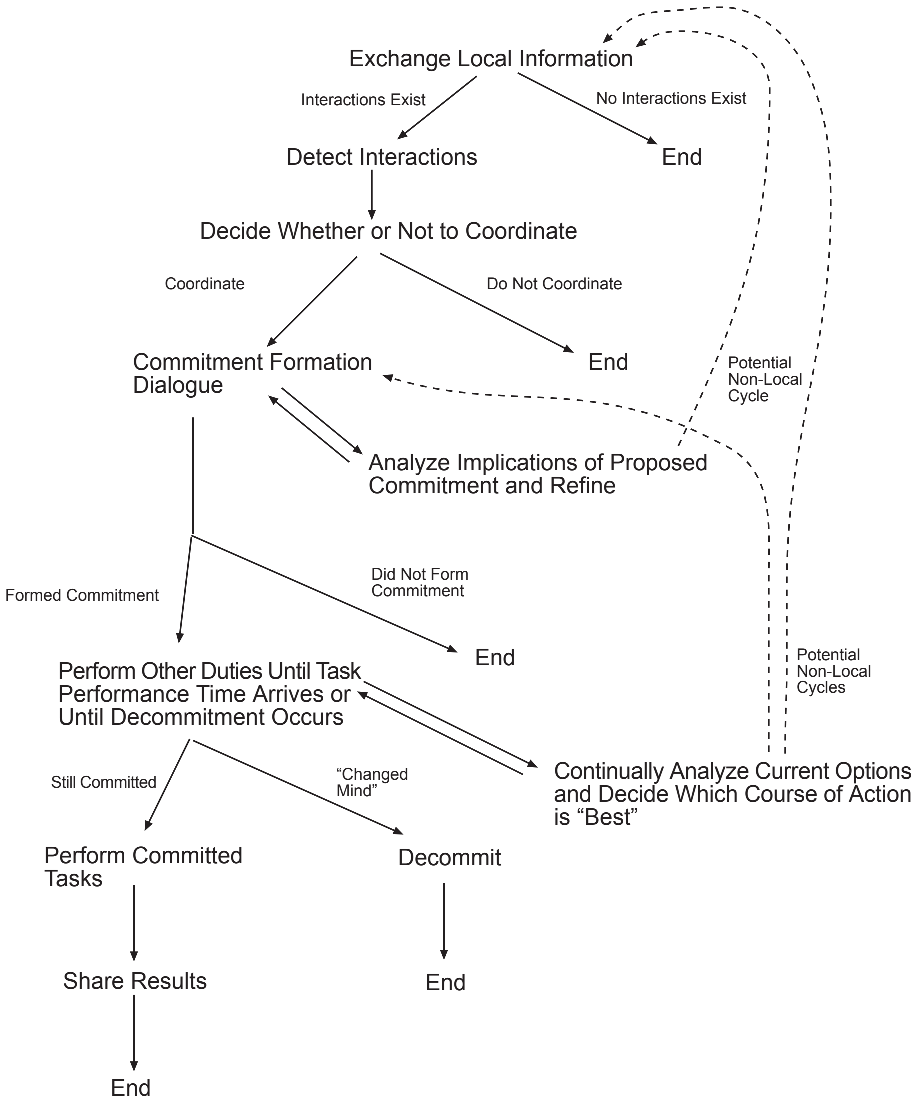

Exchange Local Information
Interactions Exist
No Interactions Exist
Detect Interactions
End
Decide Whether or Not to Coordinate
Coordinate
Do Not Coordinate
Commitment Formation Dialogue
End
Potential Non-Local Cycle
Analyze Implications of Proposed Commitment and Refine
Did Not Form Commitment
Formed Commitment
End
Potential Non-Local Cycles
Perform Other Duties Until Task Performance Time Arrives or Until Decommitment Occurs
Continually Analyze Current Options and Decide Which Course of Action is “Best”
Still Committed
“Changed Mind”
Perform Committed Tasks
Decommit
Share Results
End
End

Figure 3.1: The Coordination Process

Commitment formation and negotiation Once the decision has been made locally to coordinate, the local agent then enters a phase of commitment proposal and exchange with the other involved agents. The exact sequence of messages is dependent on the protocol used, but, this process is always present in a coordination episode [3] since commitments are necessary for agents to coordinate, as discussed previously. During the commitment formation process the local agent analyzes proposed commitments to understand the ramifications to its current set of commitments and current set of scheduled actions, and decides whether to refine the commitment (proposing a change to the commitment) or to accept/reject the commitment – agents must *decide* what to do, by *analyzing* their options from a *quantitative* perspective at all times.

Decommitment After forming a commitment, it is possible that the agent will decide to break the commitment because new opportunities have arisen or the problem solving context has changed (e.g., due to a failure, the agent can no longer perform the action that it committed to doing). If the agent decides to break a commitment, it may enter into a decommitment process with the involved agents. Again, the exact mechanics of this are dependent on the protocol – in some cases it may be appropriate for the agent to simply broadcast that it will be unable to fulfill the commitment and in other cases, perhaps where there is a variable penalty for decommitment, it may be appropriate for the agent to enter into a negotiation with the involved parties to set the decommitment penalty.

Intermediate and final results sharing Often commitments originate with the need to share results between agents. When results are produced, the producer agent must convey the required information to the consumer agent. Again, the mechanics of how this occurs are dependent on the protocol. In some cases, agents will provide status updates and intermediate results and in some situations the consumer agent may query the producer to request updates and intermediate results.

In the original GPGP work, certain aspects of these coordination-related processes are called *mechanisms*. The identification and characterization of the processes, and the separation of the processes into individually applicable mechanisms, is one of the major contributions of GPGP. In the GPGP world view, coordination consists of five mechanisms:

Update non-local viewpoints Analogous to the detection process described above.

Communicate results Analogous to the intermediate and final results sharing above.

Handle simple redundancy A mechanism that selects one agent from a set of agents who all share a common task. In GPGP, the decision aspect of this mechanism is very simple (random), however, the mechanism could be designed to use load metrics and other measures to make the determination.

Handle hard coordination relationships A mechanism that involves both commitment formation and analysis and decision making. When a hard relationship is detected, the originator of the relationship decides if and when it can provide a result to the recipient of the relationship.

Handle soft coordination relationships Identical to the above, except that this mechanism deals with "optional" relationships rather than hard (required) relationships.

[3] Unless the agents have learned from previous interactions of this type or by an off-line *a priori* specification – in which case the agent has, in essence, predetermined the outcome of the commitment formation phase. In other words, it is possible in some cases, for the commitment formation phase to occur off-line, in advance, or for the outcome of the phase to be specified in advance. In these cases, the agent does not actually need to communicate to form the commitments, but, conceptually, commitment formation is necessary in order for the agent to coordinate with the other agents at all.

These GPGP coordination mechanisms, all of which deal with some form of interaction, span the boundaries of several of the processes above. The difference in grouping stems from the difference of perspective – a GPGP coordination mechanism for handling interactions encapsulates analysis and decision making processes, commitment formation, and negotiation processes. Our recent work includes an attempt to separate and differentiate the different components of the coordination process to improve testing, extensibility, and understanding. In $GPGP^2$, the protocol used to handle coordination is separate from the analysis support code. Thus we can easily experiment with different coordination protocols, perhaps contrasting involved negotiations to simple stochastic approaches, and we can support different protocols for different problem solving contexts. For example, in a time limited context it might be appropriate to use a random selection technique to decide which agent should handle a redundant task. One of the contributions of this thesis is in the separation of the quantitative analysis framework from the coordination activity. The analysis tools and knowledge structures serve to support reasoning about activities, and determining the contextually dependent *value of tasks*, however, this is separate-able from the mechanics of coordination. In other words, deciding which possible courses of action yield the highest utility is somewhat independent of the exact mechanics of the coordination process, e.g., the exact exchange of messages. While the knowledge structuring is independent from the details of coordination, the analysis process is interdependent with the coordination process. They are related if resources consumed by the coordination process are factored-in to the decision process. In this case, the exact way in which agents will go about forming commitments affects the decision process. They are also related in that the decision process determines which actions are most important, i.e., the decision process motivates coordination. However, if we do not factor-in the resources consumed during coordination, or do so from a very gross perspective, the inter-dependence no longer holds and the decision process simply motivates coordination. Independent or interdependent? Since certain aspects of this issue are yet unsolved, the jury is still out. Part of the thesis will be an attempt to answer this question.

3.2 Task Modeling

Recall John and Al; if one were to examine the tasks assigned to these gentlemen, one would find a common factor across the different examples – the tasks assigned to (or selected by) John and Al interact. That is, the actions chosen by one of the humans affects the situation of the other. This actually seems trivial and obvious in the human context because we are almost always *socially situated*, i.e., almost always work and interact with other humans. However, the notion of tasks interacting with one another in quantifiable ways, in a computing context, was fairly long in coming. As mentioned, multi-agent coordination is about managing interdependencies between tasks belonging to different agents. If there are no interdependencies, no interactions, then there is no reason to coordinate. In the first human example, if Al did not have an errand that involved getting in the car and going somewhere, Al and John would not have a reason to coordinate.[4]

The issue of task interactions as a requisite for the need to coordinate is important because we have to be able to represent and reason about task interactions in order to coordinate over them. In our work, we generally abstract away from the agent's candidate activities and model them in a domain independent framework called TÆMS(pronounced t-e-m-s).

Figure 3.2 shows a simplified TÆMS task structure for gathering information on the Nissan Maxima via the web. The top-level task is to gather the information. The task has two subtasks which are to *Gather-Reviews* and *Find-Invoice-Price-Data*. This embodies the idea that the process of gathering purchase information on the Maxima is composed of smaller steps, gathering reviews and finding invoice price data. These steps, in turn, are also composed of smaller steps. (This is called *hierarchical decomposition*.) The decomposition ends or bottoms-out at the leaves of the tree with the square nodes, which are primitive actions. As mentioned earlier, primitive actions are generally actions

[4] Unless John was trying to get Al to do his errand for him. However, this is a case of task allocation or contracting and the interaction is through a high-level goal belonging to Al, e.g., "listen to neighbor and then respond." If Al should decide to coordinate, then the interaction might move lower in the hierarchy and be through a goal like "earn money" or "earn good will" or "help thy neighbor."

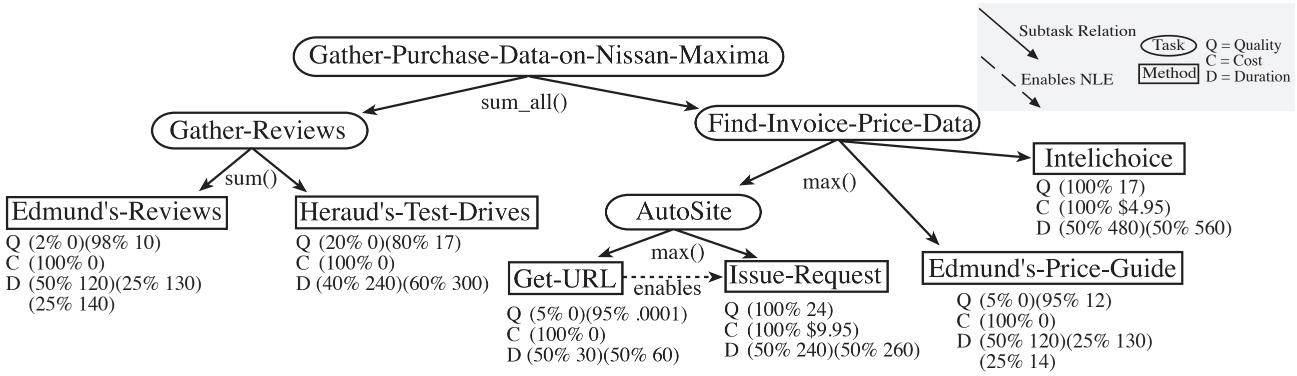


Figure 3.2: Sample TÆMS Task Structure

which cannot be decomposed any further – they are operations that the agent knows how to do that do not require any closer examination. In TÆMS, primitive actions are characterized statistically via discrete probability distributions in terms of quality, cost, and duration. (The primitive actions in the figure are annotated with these distributions.) Quality is a deliberately abstract domain-independent concept that describes the contribution of a particular action to the overall problem solving objective. Thus, different applications have different notions of what corresponds to model quality. Duration describes the amount of time that an action will take to execute. Cost describes the financial or opportunity cost inherent in performing the action modeled by the action. The statistical characteristics of the three dimensions are described via discrete probability distributions associated with each action, thus actions are actually described in terms of four dimensions: quality, cost, duration, and *uncertainty*. Because of the explicit representation of the uncertainty associated with the actions, TÆMS-based reasoners can analyze the uncertainty of certain courses of action and even take steps to reduce the uncertainty (perhaps by performing multiple redundant actions to improve the likelihood of obtaining the desired result).

The distributions associated with the actions represent *a priori* expectations about the performance characteristics of the actions. It is important to note that these expectations need not be precise specifications. Decision making with TÆMS is usually interleaved with execution and monitoring; the distribution mechanism serves as a vehicle to express expectations that are used to consider trade-offs of different possible courses of action, in much the same way that a human problem solver would use expectations to make choices.

The annotations under each of the tasks, e.g., *sum_all()*, are called quality-accumulation-functions (qafs) and they describe how the subtasks contribute to the performance of the parent task. In other words, the functions specify the number or combination of subtasks that may be performed and how the results produced by these subtasks relate to accomplishing the higher-level task or goal. The *sum_all()* qaf indicates that all of the child tasks must be performed and that the quality produced by the child tasks is summed at the parent node. In this example, the quality of the *Gather-Purchase-Data-on-Nissan-Maxima* task is computed by summing the qualities produced by *Gather-Reviews* and *Find-Invoice-Price-Data*, both of which must be performed. Qafs denote choice and they describe the relationship between the work done to perform the child task and the value or quality received at the parent task.

The *Gather-Reviews* task has two methods, query *Edmund's-Reviews* and query *Heraud's-Test-Drives*. These methods are governed by a *sum()* qaf thus the power-set of the methods minus the empty set may be performed to achieve the tasks, i.e., *Edmund's* may be queried, *Heraud's* may be queried, or both may be queried. The *Find-Invoice-Price-Data* task has three subtasks, two of type method and one of type task, governed by the *max()* qaf which is analogous to an OR relationship. Note the decomposition of the obtain invoice via *AutoSite* task into two methods, one that locates the URL and one that issues the query.

The *enables* arc leading from the *Get-URL* task to the *Issue-Request* denotes a task interrelationship; it indicates that the results of *Get-URL* are a prerequisite for *Issue-Request*. Task interrelationships in TÆMS are called *non-local-effects* (NLEs) in keeping with the notion that a local task is affected by some other (non-local) task. NLEs between tasks belonging to the same agent are still called non-local-effects because even though they are local to the agent, the effects are not contained within the single task. All NLEs in TÆMS are characterized statistically, akin to the manner in which the primitive actions are characterized. However, in the case of NLEs, the probability distributions describe the effects to quality, cost, and duration, rather than absolute quality, cost, and duration characteristics as with the primitive actions. In most cases, NLEs contain multiplier power distributions that describe the effects that they have on affected actions, i.e., the recipient action. Enables is an example of a *hard* NLE; TÆMS also models soft interactions like facilitation[5]

Back to the figure, note the low-level of quality associated with the URL finding action, this indicates that finding the URL is necessary for task achievement but that it contributes very little to achieving the task relative to the method that actually obtains the pricing report.

Modeling, structuring, and representing quantifiable effects and interactions is a powerful tool that gives TÆMS-based reasoners the ability to consider the relative value of different possible courses of action. One contribution area of this thesis will be in the extension of this knowledge representation to organizational constructs. To understand the issue of choice embodied in a task structure and its explicit representations of alternative ways to perform the top level task, consider Figure 3.3. The figure shows a set of different schedules, produced by the Design-to-Criteria scheduler (a TÆMS-based reasoner), for the task structure in Figure 3.2. Each schedule represents one different course of action for the agent, i.e., one way in which the agent may go about performing the task. Note that the schedules have different performance characteristics – in this case they were constructed for different clients having different search requirements. Schedule A is constructed for a client interested in a fast free solution with any non-zero quality. Schedule B suits a client who wants a timely and free solution, but wants less uncertainty about the expected quality of the results. Schedule C is constructed for a user interested in a good quality, free, solution that can be obtained while she goes for a cup of coffee. Schedule D is generated to meet the criteria of a fourth individual who is willing to pay and wait for a high-quality response.

Schedule A: Fast and Free

Edmund's-Reviews	Edmund's-Price-Guide

Q (~0% 0)(5% 10)(2% 12)(93% 22)
C (100% 0)
D (25% 240)(25% 250)(31% 260)(12% 270)(6% 280)

Expected Q: 21	Q Certainty: 93%
Expected C: 0	C Certainty: 100%
Expected D: 255 seconds	D Certainty: 50%

Schedule B: High Quality Certainty, Moderate Cost

Edmund's-Reviews	Intelichoice

Q (2% 17)(98% 27)
C (100% $4.95)
D (25% 600)(12% 620)(31% 680)(19% 700)

Expected Q: 26	Q Certainty: 98%
Expected C: $4.95	C Certainty: 100%
Expected D: 647 seconds	D Certainty: 50%

Schedule C: Good Quality, Moderate Cost, Slow

Edmund's-Reviews	Heraud's-Test-Drives	Intelichoice

Q (~0% 17)(20% 27)(2% 34)(78% 44)
C (100% $4.95)
D (20% 840)(19% 900)(31% 920)(19% 980)(11% 1000)

Expected Q: 40	Q Certainty: 78%
Expected C: $4.95	C Certainty: 100%
Expected D: 920 seconds	D Certainty: 70%

Schedule D: High Quality, High Cost, Moderate Duration

Edmund's-Reviews	Heraud's-Test-Drives	Get-AutoSite-URL	Issue-AutoSite-Request

Q (1% 0)(4% 27)(19% 34)(2% 41)(74% 51)
C (100% $9.95)
D (20% 630)(31% 690)(24% 720)(19% 740)(6% 760)

Expected Q: 46	Q Certainty: 74%
Expected C: $9.95	C Certainty: 100%
Expected D: 698 seconds	D Certainty: 51%

Figure 3.3: Four Satisficing Schedules

Let us now consider a simplified multi-agent scenario modeled in TÆMS, shown in Figure 3.4. In this scenario two agents are both working in the same yard. One agent has been instructed to remove a dead tree, the other agent

[5] Facilitation can be used to model a case where a result from one action can improve the ability to do some other action, by possibly increasing the resulting quality or lowering the cost or duration for the affected action.

has been assigned the task of mulching the flower beds. If they coordinate and Agent A (who is disposing of the dead tree) chips and grinds the tree first, then Agent B can use the mulch generated by this process to mulch the beds. This relationship is indicated by the *facilitates* arcs leading from the *Chip-limbs* and *Grind-stump* tasks of Agent A to the *Obtain-mulch* task of Agent B.

If the agents do not coordinate, Agent B will have to obtain the mulch from elsewhere. Note, without coordination of any form, Agent B may not even know that Agent A is going to generate mulch – the process of *discovering* interrelationships between tasks is also part of coordination. If Agent B already knew what Agent A was doing, then Agent B could "wait around" for the unidentified time at which Agent A would produce the mulch.

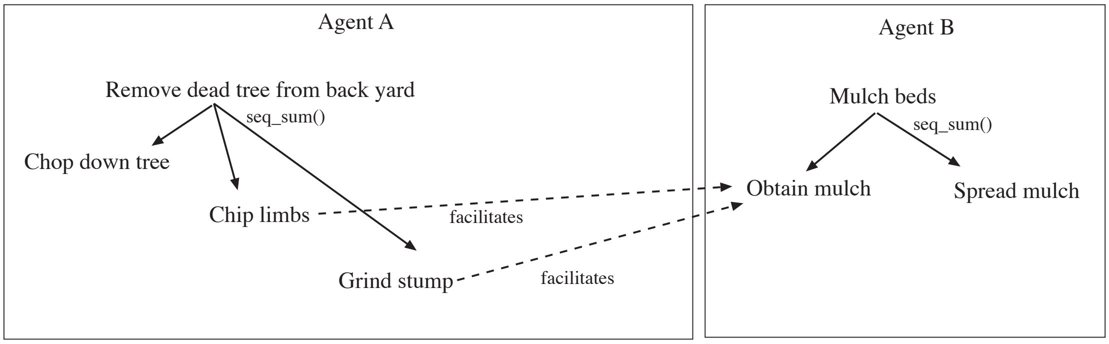


Figure 3.4: Two Agents with Interrelated Task Structures

This simple example provides a high-level view of the coordination process. First, agents must discover interactions between their tasks. They do this by comparing the items required as input and produced as output of their tasks and matching them up. After the relationships are discovered, the agents must *negotiate* to determine who is to perform what, and when. Once this determination is made, commitments are exchanged that specify which tasks a given agent will perform and the time at which they will be performed. In this example, Agent A would give a commitment to Agent B that specifies when the mulch producing actions will be performed. Agent B then arranges its activities around this commitment – in this case waiting until Agent A has produced the mulch. In this scenario, Agent B would hopefully have other tasks to perform in the meantime, however, if the agent did not have any other work to do, it may still elect to wait for the mulch produced by Agent A. The choice of whether to wait or to use some other means to obtain the mulch is dependent on the context. In a time constrained situation, Agent B may elect to go out and purchase mulch rather than wait for Agent A. The choice is value or utility driven and context dependent.

There is something lacking from this example, and even more complex and realistic examples of this form – there is no notion of belonging to an organization of agents or how organizational roles assigned to agents factor into the process of deciding what to do or deciding what is valuable, nor is it factored into decisions about when to do particular actions or into the commitment formation process. We will return to this issue in the next section; it is one of the central aspects of this thesis. The point of these examples and figures is that we must explicitly reason about the activities that the agents may carry out, and that we generally model them in a generic framework rather than working in a representational structure specific to a particular application. This enables our technology to be applied to many different applications – as long as the application-specific representation can be translated into TÆMS. With respect to TÆMS itself, it is somewhat unique in its quantification of tasks. This thesis will, by necessity, extend the use of statistical quantification in TÆMSto support organizational structures and quantified notions of commitments.

3.3 Leaving the Executive View

Obviously, an exhaustive exploration into agents and multi-agent systems is beyond the scope of this work. However, we hope that this chapter has provided enough background information to facilitate understanding of the thesis topic and the planned (and ongoing) contributions to multi-agent systems.

Chapter 4

Where We Are

Chapter 3 discusses work in multi-agent systems and relates it to the thesis topic, which is presented in Chapter 2. In this chapter we explore current local research in multi-agent systems and relate it to the proposed topic and research path. This is highly relevant as the thesis research builds on a foundation provided by a long history of successful research in distributed problem solving and multi-agent agent systems. More importantly, certain aspects of the thesis research entail extending and enhancing, in significant ways, knowledge structures and reasoning systems currently employed by the local research community. Chapter 5 returns to the topic directly, Chapter 6 discusses highly related work in detail, and Chapter 7 presents a research plan and sketches an evaluation plan.

4.1 Agent Architecture

Figure 4.1 shows a detailed view of our prototypical agent architecture. Within the complex architecture there are three main areas of interest, each of which is a locus of control. The different loci are: the domain problem solver, the local agent scheduler, and the coordination mechanism. These three components appear separately in Figure 4.2. Each of the primary components perform a specific problem-solving function, however, there are actually only two classes of problem solving performed by the three components, *domain* problem solving and *control* problem solving. *Domain problem solving* denotes work spent determining which primitive actions could be used to achieve a domain goal, performing the actions, and then evaluating the results of the actions to determine how the results affect the planned course of action. In contrast to these domain-centric activities, the term *control problem solving* denotes work spent deciding a course of action from a set of candidate actions, in what sequence to perform the actions, and coordinating with other agents. Note, these are both types of problem solving behaviors, however, the area of specialization differs. Domain problem solvers reason about some area of expertise like network diagnoses or Windows98 troubleshooting. Control problem solvers reason about control – their area of expertise, if you will, is in controlling the agent.

Thus, the domain problem solver in Figure 4.2 is the application expert. For example, in our information gathering agent [60], the domain problem solver is an interpretation-style planner that “knows about” gathering information on the web to make product purchase decisions. The domain problem solver could be a standard generative planner, a process program, or even a human expert. The domain problem solver is responsible for reasoning about the activities required to carry out the application, e.g., mow the lawn, gather information, design software, etc. The problem solver abstracts its problem solving options (process plans, design steps, hierarchical plans, etc.) into a TÆMS task structure which is then sent to the local agent scheduler and the coordination module for analysis.

Unlike the domain problem solver, the local agent scheduler is a control problem solver, not a domain expert. It is an expert at reasoning about the constraints between the different possible options and the options’ different quality,

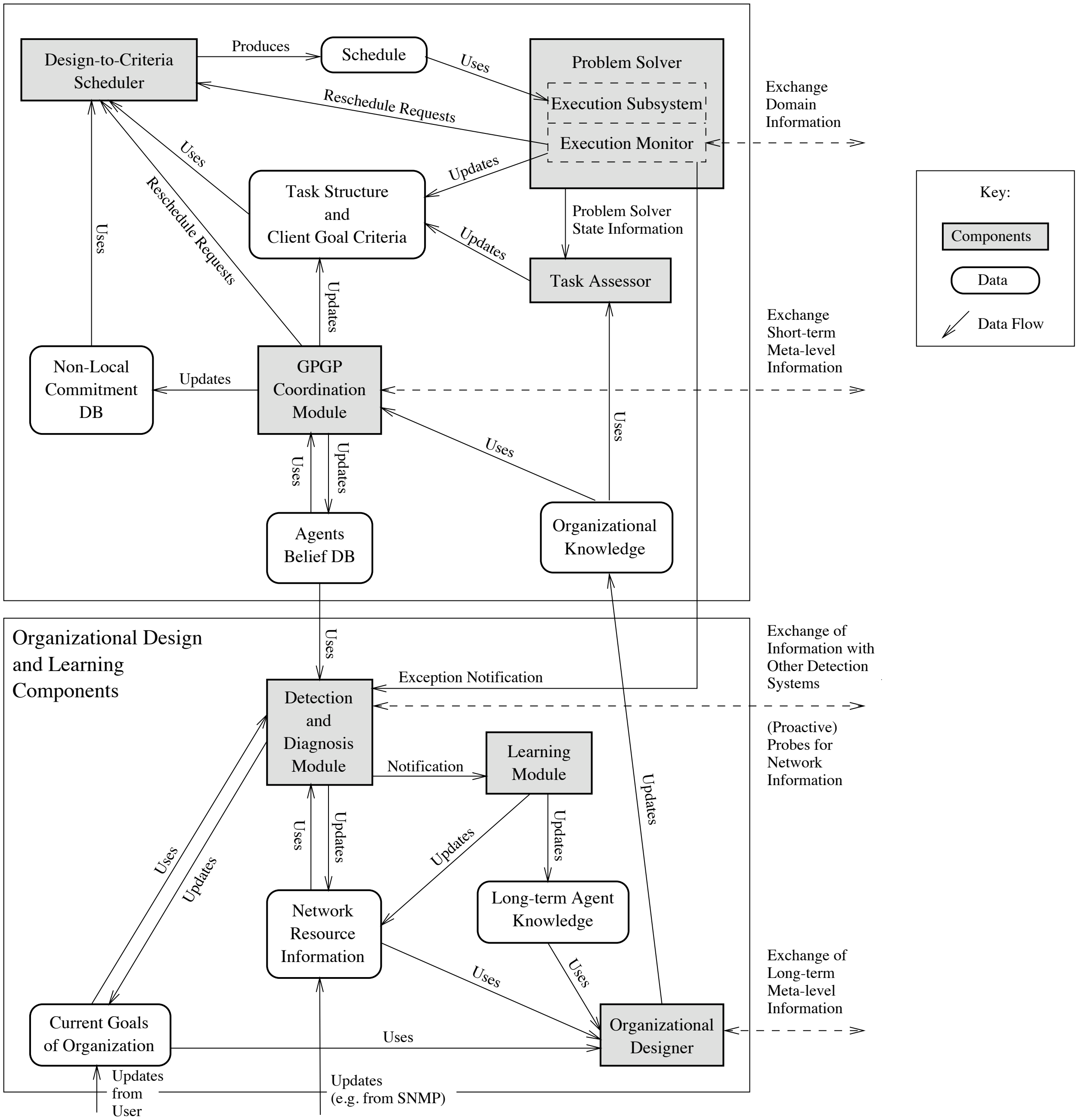

Design-to-Criteria Scheduler
Produces
Schedule
Uses
Problem Solver
Execution Subsystem
Execution Monitor
Reschedule Requests
Exchange Domain Information
Uses
Reschedule Requests
Updates
Task Structure and Client Goal Criteria
Updates
Problem Solver State Information
Task Assessor
Uses
Updates
Key:
Components
Data
Data Flow
Exchange Short-term Meta-level Information
Non-Local Commitment DB
Updates
GPGP Coordination Module
Uses
Uses
Updates
Uses
Agents Belief DB
Organizational Knowledge
Organizational Design and Learning Components
Uses
Exception Notification
Exchange of Information with Other Detection Systems
Detection and Diagnosis Module
Notification
Learning Module
(Proactive) Probes for Network Information
Updates
Uses
Updates
Updates
Updates
Uses
Updates
Network Resource Information
Long-term Agent Knowledge
Uses
Uses
Exchange of Long-term Meta-level Information
Current Goals of Organization
Uses
Organizational Designer
Updates from User
Updates (e.g. from SNMP)

Figure 4.1: Agent Architecture – Full View

cost, duration, and uncertainty trade-offs. The scheduler is a domain-independent component that can schedule or reason about any type of application or process as long as it is described via a (domain independent) TÆMS task model. It is the job of the domain problem solver to enumerate its options and to carry these out, but it is the job of the scheduler to analyze the alternative different actions that the problem solver could carry out and to select a set of these, and sequence them, in order to maximize the overall utility (per some utility specification in terms of quality, cost, duration, and uncertainty). The current incarnation of our scheduling component is call the Design-to-Criteria scheduler and we will discuss its decision process (since it relates directly to the thesis direction) in Section 4.2.

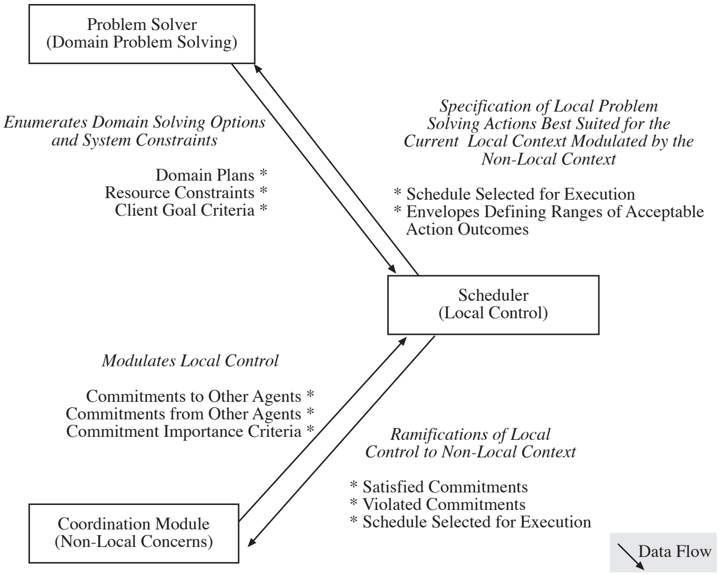

Problem Solver
(Domain Problem Solving)
Enumerates Domain Solving Options
and System Constraints
Domain Plans *
Resource Constraints *
Client Goal Criteria *
Specification of Local Problem
Solving Actions Best Suited for the
Current Local Context Modulated by the
Non-Local Context
* Schedule Selected for Execution
* Envelopes Defining Ranges of Acceptable
Action Outcomes
Scheduler
(Local Control)
Modulates Local Control
Commitments to Other Agents *
Commitments from Other Agents *
Commitment Importance Criteria *
Ramifications of Local
Control to Non-Local Context
* Satisfied Commitments
* Violated Commitments
* Schedule Selected for Execution
Coordination Module
(Non-Local Concerns)
Data Flow

Figure 4.2: Simple Agent Architecture

The third component is the *coordination module*, historically this has been GPGP in our work, though we are currently working on a replacement (aka $GPGP^2$). If an agent operates in an environment where its tasks are entirely independent of the tasks belonging to other agents, or where it is the only agent in the environment (e.g., deep space, deep ocean), then the agent would have little need for a coordination module. However, in a multi-agent system, agents do not operate in a vacuum and decisions about which actions to perform must include considering how the tasks and goals of the other agents in the system affect the local situation, and vice versa. The coordination module is the interface by which agents coordinate their activities, and through which the local agent scheduler gleans information about the multi-agent system as a whole. In our view, the larger context does not necessarily supersede the local goals and tasks of a given agent, i.e., coordination does not dominate the local agent's actions. Instead, coordination *modulates* local activities to various degrees depending on the types and effects of the interactions between agents.

One of the main contributions of this specialization-based agent architecture is its attempt to approach the problem from a domain-independent perspective. Both the Design-to-Criteria (DTC) scheduler and the GPGP coordination module understand the application through the use of the TÆMS modeling framework. Any activity that can be appropriately modeled in TÆMS is thus a problem instance on which these tools can operate. The domain-independence of TÆMS coupled with the separation of domain and control problem solving mean that the control problem solving components can be bundled with any domain problem solver in order to construct multi-agent systems. The only significant requirements are that the problem solver be able to translate its process into TÆMS and that it be able to delegate control decisions to the scheduler. Domain-independence is an implicit goal of this thesis work – organizational concerns and valuation issues must be approached in a domain independent fashion.

Thus, the domain problem solver and the two control problem solvers interact (via TÆMS) to control the agent and to achieve the agent's assigned tasks or goals. While agent control may be effective, it is probably far from optimal. The scheduler makes decisions about what to do, and when, based on a limited view of the agent's activities. The agent models and reasons about *only* its candidate domain problem solving actions – it does not explicitly consider coordination related actions. For example, the work required to send a result to another agent is not explicitly considered by the scheduler. In some cases, it may actually be beneficial to the agent to wait to send a result, or to not send a result at all (in a time constrained situation, for instance). Additionally, when it evaluates how its own actions relate to those of other agents, it does so from a very second hand perspective. It does not reason about some overall value or utility for the actions, it only reasons about their quality, cost, duration, and uncertainty characteristics in light of a goal specification that is in terms of the domain tasks or goals, i.e., multi-agent considerations are an add-on to the agent's decision process (and cast in terms of its domain problem solving activities).

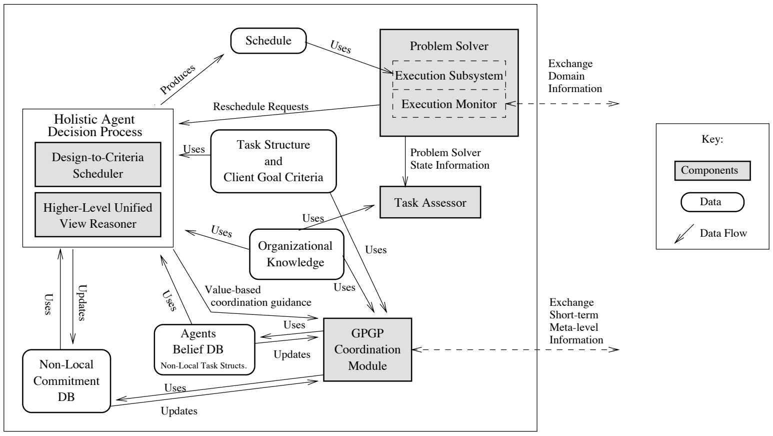


Figure 4.3: Modified View of the Primary Agent Components and Thier Interactions

While our work is currently state-of-the-art, we have come to understand that explicit representation of non-local concerns and first class analysis of these is necessary for scaling up multi-agent systems to complex real-world environments and applications. This is the central issue of this dissertation. Agents must represent and consider the larger problem solving context. Local control (scheduling) should still be influenced by coordination activities, but the degree of the influence should be contextually dependent, and the context needs to be expanded to include notions like organizational roles, the value of actions to an organization, the benefit of cooperating with other agents in an organization, etc. With the integration of the Design-to-Criteria scheduler and the new organizationally centered decision component, the architecture pictured in the previous figures will change, probably as shown in Figure 4.3. As discussed in Section 2.4, the new decision component will replace some of the decisions currently handled by the coordination module; in the full view agent architecture (Figure 4.1) organizational knowledge will also be used directly by the decision component (embodied in this figure by the Design-to-Criteria scheduler). In the new work, the decision component will also be responsible for guiding coordination (*modulating* coordination), not simply responding to or being modulated by, coordination and the coordination module will cease to actively manage the local decision process.

4.2 Design-to-Criteria Scheduling

As part of our agent control work and as a contribution to this thesis, we have developed a domain independent flexible computation [45, 18, 46, 68, 95, 96] approach to task scheduling called *Design-to-Criteria*. The most distinguishing features of Design-to-Criteria (DTC) are the ability to reason about the utility attribute trade-offs of different solutions based on different goal criteria, the ability to use these utility attribute trade-offs to focus every step of the scheduling process, and the ability to do these activities from a satisficing[1] perspective. Satisficing with respect to the scheduling process itself enables the scheduler to produce results when computational combinatorics prevent an optimal solution. Satisficing with respect to meeting the goal criteria enables the scheduler to produce a result that adheres to the spirit of the goal when the criteria cannot be satisfied perfectly due to environmental and resource constraints.

The DTC scheduler is the agent's local expert on making control decisions. The scheduler's role is to consider the possible domain actions enumerated by the domain problem solver and choose a course of action that best addresses: 1) the local agent's goal criteria (its preferences for certain types of solutions), 2) the local agent's resource constraints and environmental circumstances, and 3) the non-local considerations expressed by the GPGP coordination module. The general idea is to evaluate the options in light of constraints and preferences from many different sources and to find a way to achieve the selected tasks that best addresses all of these. The scheduler's decision process is central to this thesis research because the new organizational knowledge and wider context all pertain to this decision process.

It is important to realize that the decision process is ongoing. Schedules are not typically produced and then executed in a complete end-to-end fashion. During normal agent execution, the scheduler is typically invoked multiple times to respond to changes in the problem solving context. Factors contributing to this behavior include: environmental uncertainty; environmental dynamism; the approximation techniques used in scheduling, coordination, and domain problem solving; the potential of learning new information that impacts the problem solving process. Efficiency is a major scheduler requirement because the scheduling decision process occurs frequently and because we are interested in operating in "real-world" domains where resources are limited and deadlines are common place.

In the domain-independent agent architecture, Figures 4.1 and 4.2, the primary components (problem solver, the GPGP coordination module, and the DTC scheduler) exchange information using the TÆMS task modeling framework. The scheduler's problem is thus framed in terms of a TÆMS task structure emitted by the domain problem solver. Our agent control research focuses on a class of problem solving processes where there are typically multiple different actions for performing a particular task, each action has different statistical performance characteristics, and uncertainty about the outcomes of actions is ubiquitous. For example, in the signal processing domain [55] there are multiple different techniques that can be used to process and identify signals; an approximate signal processing algorithm such as QSTFT (quantized short-time Fourier transform) [63] is inexpensive to compute but likely to produce interpretations that have significant uncertainty and there is a high probability that the interpretations will altogether miss certain types of signal sources. In contrast, a STFT (short-time Fourier transform) [64] is expensive to compute, but has very good quality and it is highly likely that all signal sources will be represented to some degree in the interpretation. This example is deliberately simple to illustrate a point – consider a case where there are *many* different actions for achieving a particular task and any combination of the actions can be employed and possibly in any order. Now consider a hierarchy of such tasks where the tasks themselves are interrelated and constrained by deadlines and resource limits. The TÆMS framework models such problem solving processes. As discussed, in TÆMS, primitive actions are modeled statistically via discrete probability distributions in three dimensions, quality, cost, and duration. Probability distributions are also associated with task interactions, called *NLEs* (non-local-effects), e.g., precedence constraints or advantageous soft relationships, and the effects of the interactions are reasoned about statistically.

Scheduling problem solving activities modeled in the TÆMS language has four major requirements: 1) to find a set of actions to achieve the high-level task, 2) to sequence the actions, 3) to find and sequence the actions in soft real-time, 4) to produce a schedule that meets dynamic goal criteria, i.e., cost, quality, duration, and certainty requirements,

[1] Meaning "approximate" or the "best-that-is-possible" given the combinatorics

of different clients. TÆMS models multiple approaches for achieving tasks along with the quality, cost, and duration characteristics of the primitive actions, specifically to enable TÆMS clients to reason about the trade-offs of different courses of action. In other words, for a given TÆMS task model, there are multiple approaches for achieving the high-level task and each approach has different quality, cost, duration, and certainty characteristics. In contrast to classic scheduling problems, the TÆMS scheduling objective is not to sequence a set of unordered actions but to *find and sequence* a set of actions that *best* suits a particular client's quality, cost, duration, and certainty needs. Design-to-Criteria is about examining the current situation, the current options before the agent, and deciding on a course of action – it is about targetable contextual decision making. These ideas are the cornerstone of this thesis direction. In this thesis, we propose to extend the scope and possibly produce new non-TÆMS structures for analysis, and possibly even a whole new analysis process, but the intellectual ideas of the DTC approach are likely to contribute to this work.

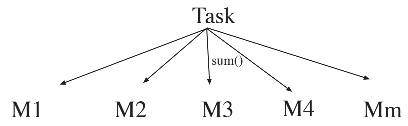


Figure 4.4: A Very Simple Task Structure

Design-to-Criteria scheduling requires a sophisticated heuristic approach because of the scheduling task's inherent computational complexity. To understand the complexity and get a feel for the scheduling process, consider the task structure shown in Figure 4.4. It is only a single level deep – the single task has m child methods (primitive actions) and it accumulates quality according to the `sum()` qaf. In this case, there are $2^m - 1$ unordered sets of methods that can be used to achieve the parent task, and within each set of n methods, $n!$ possible orderings of methods in the schedule. These complexities are associated with two different aspects of the TÆMS scheduling problem. The first source of complexity stems from the process of deciding which methods to use to achieve a particular task, i.e., selecting unordered methods from a set of candidate methods. The complexity of this process is $O(2^m)$ (the power-set of the number of methods). The second source of complexity is caused by the number of possible schedules that can be created for each unordered method set - $O(n!)$[2] where n is the number of methods in a given set. Combining the two sources of complexity, the general upper-bound on the number of possible schedules for a TÆMS task structure containing m methods is given in Equation 4.1. Clearly, for any significant task structure the brute-strength approach of generating all possible schedules is infeasible.

$$\sum_{i=0}^{m} \binom{m}{i} i! \tag{4.1}$$

The $\omega(2^n)$ and $o(n^n)$ (by Stirling's approximation) combinatorics of our scheduling problem precludes using exhaustive search techniques for finding optimal schedules. Furthermore, the deadline and resource constraints on tasks, plus the existence of complex task interrelationships, prevent the use of a single heuristic for producing optimal or even "good" schedules. Design-to-Criteria copes with these explosive combinatorics by satisficing with respect to the goal criteria and with respect to searching the solution space. This satisficing dualism translates into four different techniques that Design-to-Criteria uses to reduce the search space and make the scheduling problem tractable:

Criteria-Directed Focusing The client's goal criteria is not simply used to select the "best" schedule for execution, but is also leveraged to focus all processing activities on producing solutions and partial solutions that are most

[2] The complexity of the action ordering task in TÆMS is actually $O((m * n) * (m!))$, where m is the number of actions to order and n is the number of nodes in the TÆMS task structure. The added factor is generated by the possibility of task interactions. When adding each method to the schedule, the entire task structure may have to be processed to propagate the non-local-effects and compute the effects of task interactions.

likely to meet the trade-offs and limits/thresholds defined by the criteria. This is achieved by creating and identifying partial solutions that seem likely to meet the criteria and concentrating further development on these classes of partial solutions, pruning or ignoring other partial solutions that are deemed least probable to lead to "good" solutions.

Approximation Schedule approximations, called *alternatives*, are used to provide an inexpensive, but coarse, overview of the schedule solution space. Alternatives contain a set of unordered actions that can be scheduled (ordered) to achieve a particular task along with estimates for the quality, cost, and duration distributions that may result from scheduling the actions. Alternatives are inexpensive to compute as the complex task interactions are only partially considered and ordering, resource, and other constraints are ignored. The alternative abstraction space is used in conjunction with the criteria directed focusing to build schedules from alternatives that are most likely to lead to good schedules. This helps to reduce the $O(2^m)$ class of complexity by reducing the number of unordered method sets that the agent considers for scheduling from $O(2^m)$ to some fixed upper bound.

Heuristic Decision Making Given a set of n actions to perform there are $n!$ orderings that must be considered and the $O(n!)$ expense is non-trivial. We cope with this complexity using a group of heuristics for action ordering. The heuristics take into consideration task interactions, attempting to take advantage of positive interactions while avoiding negative interactions. They also consider resource limits, individual action deadlines, task deadlines, commitments made with other problem solving agents, and other constraints. The heuristic algorithm reduces the $O(n!)$ action ordering problem to low-order polynomial levels in the worst case.

Heuristic Error Correction The use of approximation and heuristic decision making has a price – it is possible to create schedules that do not achieve the high-level task, or, achieve the high-level task but do not live up to quality, cost, duration, or certainty expectations set by the estimates contained in the alternatives. This can be caused by an overconstrained problem, but also by complex task interactions that are glossed over by the alternative approximation and not considered by the action ordering heuristics. A secondary set of improvement [97] heuristics act as a safety net to catch the errors that are correctable. Again, this problem is potentially computationally expensive as the required fix may be achievable by any combination of the actions in the task structure and it is impossible to ascertain if a hypothetical fix will generate the desired result until it is fully scheduled. Thus this aspect of the scheduling algorithm is also heuristic and relies on abstraction and criteria directed focusing to reduce the complexity.

Design-to-Criteria thus copes with computational complexity by using the client goal criteria to focus processing, reasoning with schedule approximations rather than complete schedules, and using a heuristic, rather than exhaustive, scheduling approach. This methodology is effective because several aspects of the scheduling problem are soft and amenable to a satisficing approach. For example, the client goal specification mechanism, Figure 4.6, expresses soft client objectives or soft constraints. Solutions do not need to meet absolute requirements because clients cannot know *a priori* what types of solutions are possible for a given task structure due to the combinatorics. Hard constraints do exist in TÆMS, but they originate from commitments entered into with other problem solvers and from the tasks themselves. Similarly, soft task interactions also represent soft constraints that can be relaxed, i.e., they can be leveraged or not depending on the situation. Finally, though the TÆMS scheduling problem is more complex than many traditional scheduling problems because of its representation of multiple approaches for task achievement, it is also more flexible. If we view the scheduling activity as a search process, typically there is a neighborhood of solutions that will meet the client's goal criteria and the lack of exhaustive search, i.e., search by focused processing and approximation, does not necessitate scheduling failure.

The overall DTC scheduling process proceeds in phases. A control-flow view of the process is shown in Figure 4.5. Prior to scheduling, the scheduler client describes the problem instance via TÆMS and expresses his/her/its preferences in terms of quality, cost, duration, certainty, and thresholds and limits on these values. A gui view of the

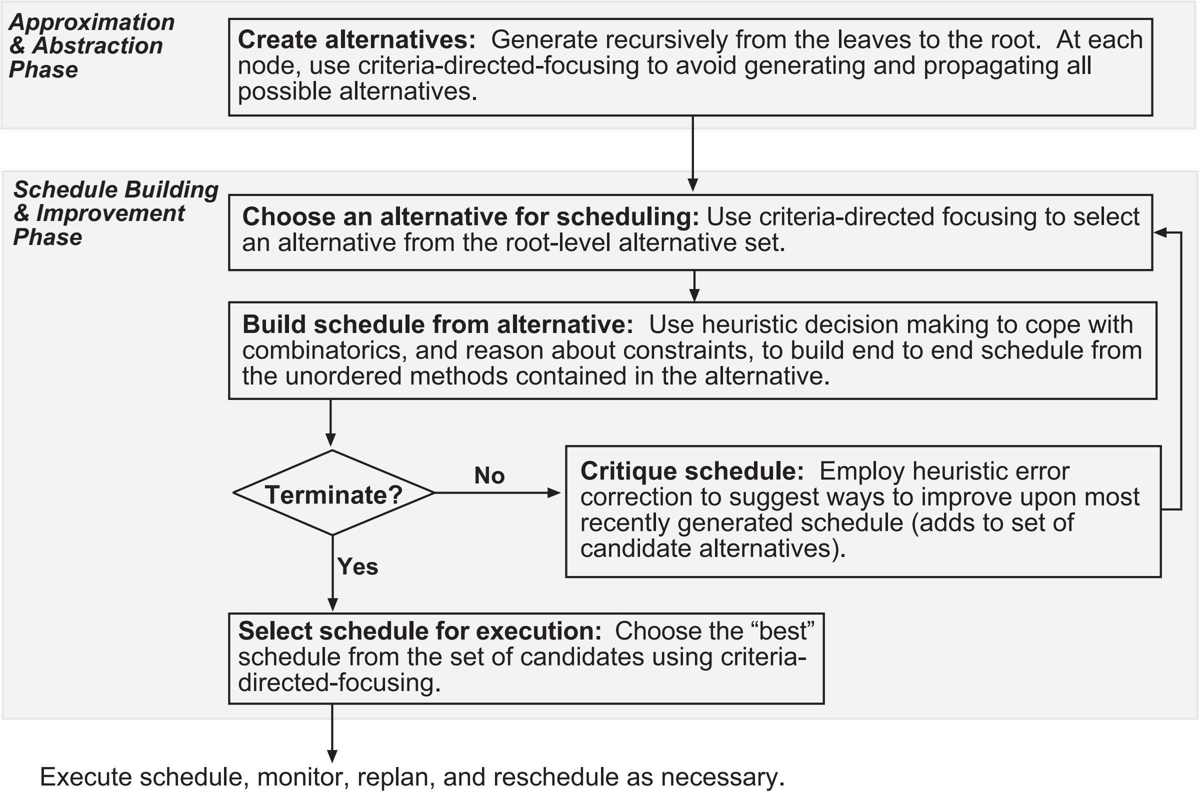

Approximation & Abstraction Phase
Create alternatives: Generate recursively from the leaves to the root. At each node, use criteria-directed-focusing to avoid generating and propagating all possible alternatives.
Schedule Building & Improvement Phase
Choose an alternative for scheduling: Use criteria-directed focusing to select an alternative from the root-level alternative set.
Build schedule from alternative: Use heuristic decision making to cope with combinatorics, and reason about constraints, to build end to end schedule from the unordered methods contained in the alternative.
Terminate?
No
Critique schedule: Employ heuristic error correction to suggest ways to improve upon most recently generated schedule (adds to set of candidate alternatives).
Yes
Select schedule for execution: Choose the "best" schedule from the set of candidates using criteria-directed-focusing.
Execute schedule, monitor, replan, and reschedule as necessary.

Figure 4.5: High-Level Control-Flow View of Design-to-Criteria

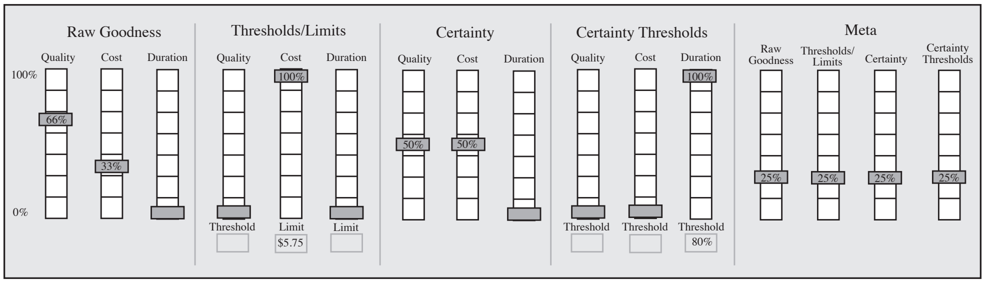


Figure 4.6: A Slider Set Describing a Particular Client's Objectives

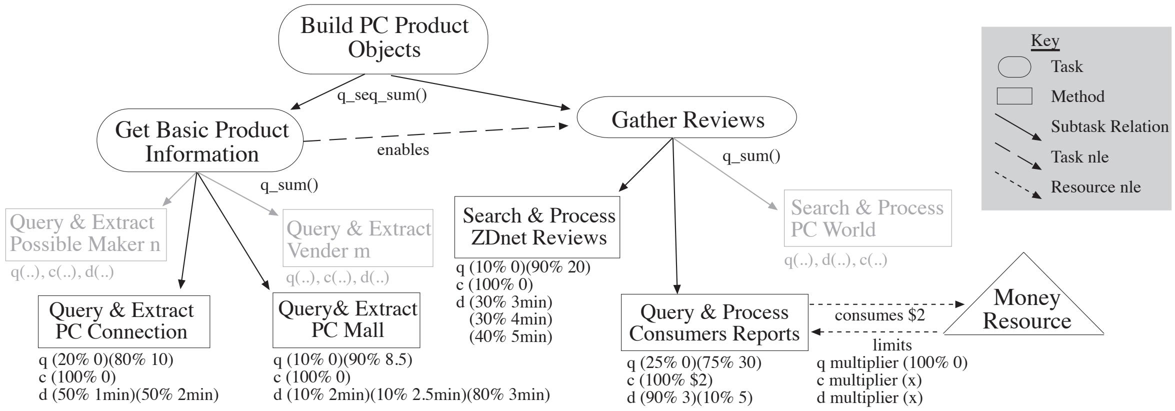


Figure 4.7: Information Gathering Task Structure

specification metaphor is given in Figure 4.6. Since Design-to-Criteria scheduling is directly related to the issue of incorporating organizational knowledge into an agent's coordination and decision processes, consider a high-level walk through of a simple scheduling episode. The input is the information gathering task structure [3] shown in Figure 4.7 (we will consider only the dark nodes for simplicity) accompanied by goal or design criteria as follows:

- **Bank 1 - Raw Goodness:** Goodness quality slider = 60%, goodness duration slider = %40 (goodness cost slider = 0%)
- **Bank 2 - Thresholds/Limits:** Threshold cost slider = 100% and a cost limit of $0. (All other sliders set to 0% and thus the thresholds and limits are *don't care* conditions.)
- **Bank 3 - Certainty:** All set to 0%.
- **Bank 4 - Certainty Thresholds:** All set to 0%.
- **Bank 5 - Meta:** Raw goodness = 50% and thresholds/limits = 50%.

The criteria describes a client who is in a hurry and broke, but who cares more about a quality result than an extremely fast result. The general utility equations are given in [83]. For this particular example, based on the criteria above, a schedule's utility is computed as follows. Let $this.eq$ denote the expected quality of the schedule under consideration; $this.ed$ is the expected duration and $this.ec$ is the expected cost. Let min_q and max_q denote the observed minimum and maximum qualities, and min_c/max_c and min_d/max_d denote the same for cost and duration. A schedule's utility is determined by evaluation in light of the two different classes of expressed criteria, namely 1) raw quality and duration characteristics, and 2) meeting the cost constraint. The two different classes produce "sub ratings" that are then weighed and merged to determine overall schedule utility as follows:

$$utility_{raw\ goodness} = \frac{(this.eq - min_q)}{max_q - min_q} * .60 + \frac{(max_d - this.ed)}{max_d - min_d} * .40$$

$$\begin{array}{l} if\ (this.ec \leq 0)\ then \\ \quad utility_{thresholds/limits} = .50 \\ else \\ \quad utility_{thresholds/limits} = 0 \end{array}$$

[3] The structure is actually a simplified fragment of a larger structure, but sufficient for illustration purposes.

$$utility \quad = utility_{raw\ goodness} * .50 + utility_{thresholds/limits} * .50$$

The first step in the scheduling process is to enumerate the different ways to go about achieving the top-level task. Each "way" is called an alternative. Alternatives are constructed from the leaf-nodes up to the top-level task node according to the qafs associated with the nodes. Both *Gather-Reviews* and *Get-Basic* have *sum()* qafs thus any combination of their subtasks may be used to achieve the parent task (the power-set of their children minus the empty set). *Build-PC-Objects* has a *seq_sum()* qaf, thus both the *Gather-Reviews* and *Get-Basic* tasks must be performed (the cross product of the alternative sets associated with these nodes). Since the task structure is very small, the entire alternative set for *Build-PC-Objects* may be constructed (otherwise, only a portion of the alternatives are propagated and constructed). The alternative set for the top level task is shown in Figure 4.8. Each alternative contains an unordered set of methods and an estimate[4] for the quality, cost, and duration that will result from building a schedule out of the methods. Each alternative is accompanied by a utility rating. This is computed in the same fashion that utilities are computed for schedules and using the same criteria. The only difference is that the distributions used to produced the expected values used in the formulas are *estimates* rather than actual schedule values.

Alternatives are turned-into schedules by ordering or sequencing the methods in the alternatives. This is analogous to a general scheduling problem, given a set of actions, find a sequence for them. However, it is complicated by the existence of interactions between the actions and individual constraints on them like deadlines, quality requirements, and so forth. As mentioned, schedules are created using a set of heuristics that reduce the exponential problem to polynomial levels. After the top-level alternative set is constructed, the scheduler then iterates, selecting the next highest rated alternative (that with the highest utility) and constructing a schedule for it. The process stops based on a complex set of halting criteria that includes: 1) no alternatives have a higher potential than the best of the produced schedules, 2) running out of alternatives to schedule, 3) running out of time for scheduling, 4) building some predefined number of schedules. Again, since the task structure in question is very small, the complete[5] schedule set for *Build-PC-Objects* can be produced and is shown in Figure 4.9. Note: much of the detail is omitted for clarity, including the distributions and statistical characterizations of each method in the schedule.

Note that the top three schedules cluster together and have a similar utility rating. This is because all three schedules address the cost-limit preference (having no cost) and obtain a good balance of quality and duration. The highest rated schedule has a quality that is significantly better than the second rated schedule, but, it also has a longer duration. In contrast, the lower-rated schedule SCHEDULE_meta_root_sum_8 obtains the highest quality, but also does so in a greater amount of time and while incurring a cost.

Relating this example back to the thesis topic – the decision process presented here is done from a unified perspective, though one with a limited scope. All of the candidate activities are related by utility, and the utility calculation is done by comparing the quality, cost, and duration of the different *domain* actions. One objective of this thesis is to broaden the scope of the decision process to include notions of organizations of agents and the importance of an agent's tasks to the overall organization. In this example, there is no notion of an organizational context. All actions are evaluated on the merits of their individual quality, cost, and duration values – values that relate *only* to the domain value of these actions, not to some overall good. We will return to this point in Chapter 5.

[4] Alternatives actually also contain potentials – high-end estimates for quality, cost, and duration that represent the best possible outcome if all soft positive interactions are leveraged by the scheduler. Since this example lacks soft positive interactions, the potential distributions are identical to the baseline distributions and omitted for clarity.

[5] Note: this is not the exhaustive set of schedules, but the entire set as produced by the scheduler's heuristic approach. This means that for each possible combination of primitive actions, having m methods, that can achieve the top-level task, there is one schedule rather than the $m!$ schedules that would result from trying all sequences of the m actions.

```
Alternative:    meta_root_sum_0
                Utility : 0.677529
                Unordered Method List : ZDnet-Reviews PC-Connection
                Estimated Quality, expected value 26.0, distribution : (0.02 0.0)(0.08 10.0)(0.18 20.0)(0.72 30.0)
                Estimated Cost, expected value 0.0, distribution : (1.00 0.0)
                Estimated Duration, expected value 5.6, distribution : (0.15 4.0)(0.30 5.0)(0.35 6.0)(0.20 7.0)
Alternative:    meta_root_sum_2
                Utility : 0.670502
                Unordered Method List : ZDnet-Reviews PC-Mall PC-Connection
                Estimated Quality, expected value 33.6, distribution : (0.00 0.0)(0.02 8.5)(0.01 10.0)(0.07 18.5)(0.02 20.0)(0.16 28.5)(0.07 30.0)(0.65 38.5)
                Estimated Cost, expected value 0.0, distribution : (1.00 0.0)
                Estimated Duration, expected value 8.4, distribution : (0.02 6.0)(0.16 7.0)(0.03 7.5)(0.28 8.0)(0.00 8.4)(0.04 8.5)(0.31 9.0)(0.17 10.0)
Alternative:    meta_root_sum_1
                Utility : 0.635245
                Unordered Method List : ZDnet-Reviews PC-Mall
                Estimated Quality, expected value 25.6, distribution : (0.01 0.0)(0.09 8.5)(0.09 20.0)(0.81 28.5)
                Estimated Cost, expected value 0.0, distribution : (1.00 0.0)
                Estimated Duration, expected value 7.0, distribution : (0.03 5.0)(0.03 5.5)(0.27 6.0)(0.03 6.5)(0.28 7.0)(0.04 7.5)(0.32 8.0)
Alternative:    meta_root_sum_6
                Utility : 0.306010
                Unordered Method List : Consumers-Reports ZDnet-Reviews PC-Connection
                Estimated Quality, expected value 48.5, distribution : (0.01 0.0)(0.02 10.0)(0.04 20.0)(0.19 30.0)(0.06 40.0)(0.13 50.0)(0.54 60.0)
                Estimated Cost, expected value 2.0, distribution : (1.00 2.0)
                Estimated Duration, expected value 8.8, distribution : (0.14 7.0)(0.27 8.0)(0.33 9.0)(0.21 10.0)(0.04 11.0)(0.02 12.0)
Alternative:    meta_root_sum_8
                Utility : 0.300000
                Unordered Method List : Consumers-Reports ZDnet-Reviews PC-Mall PC-Connection
                Estimated Quality, expected value 56.3, distribution : (0.01 0.0)(0.08 28.5)(0.18 38.5)(0.07 48.5)(0.00 56.1)(0.12 58.5)(0.05 60.0)(0.49 68.5)
                Estimated Cost, expected value 2.0, distribution : (1.00 2.0)
                Estimated Duration, expected value 11.6, distribution : (0.02 9.0)(0.16 10.0)(0.26 11.0)(0.03 11.5)(0.00 11.7)(0.30 12.0)(0.20 13.0)(0.03 15.0)
Alternative:    meta_root_sum_7
                Utility : 0.263726
                Unordered Method List : Consumers-Reports ZDnet-Reviews PC-Mall
                Estimated Quality, expected value 48.1, distribution : (0.00 0.0)(0.02 8.5)(0.02 20.0)(0.20 28.5)(0.01 30.0)(0.07 38.5)(0.07 50.0)(0.61 58.5)
                Estimated Cost, expected value 2.0, distribution : (1.00 2.0)
                Estimated Duration, expected value 10.2, distribution : (0.04 8.0)(0.27 9.0)(0.27 10.0)(0.00 10.2)(0.04 10.5)(0.32 11.0)(0.03 12.0)(0.03 13.0)
Alternative:    meta_root_sum_3
                Utility : 0.247547
                Unordered Method List : Consumers-Reports PC-Connection
                Estimated Quality, expected value 30.5, distribution : (0.05 0.0)(0.20 10.0)(0.15 30.0)(0.60 40.0)
                Estimated Cost, expected value 2.0, distribution : (1.00 2.0)
                Estimated Duration, expected value 4.7, distribution : (0.45 4.0)(0.45 5.0)(0.05 6.0)(0.05 7.0)
Alternative:    meta_root_sum_5
                Utility : 0.240520
                Unordered Method List : Consumers-Reports PC-Mall PC-Connection
                Estimated Quality, expected value 38.1, distribution : (0.01 0.0)(0.04 8.5)(0.02 10.0)(0.18 18.5)(0.02 30.0)(0.13 38.5)(0.06 40.0)(0.54 48.5)
                Estimated Cost, expected value 2.0, distribution : (1.00 2.0)
                Estimated Duration, expected value 7.6, distribution : (0.04 6.0)(0.04 6.5)(0.41 7.0)(0.04 7.5)(0.00 7.6)(0.37 8.0)(0.05 9.0)(0.04 10.0)
Alternative:    meta_root_sum_4
                Utility : 0.205262
                Unordered Method List : Consumers-Reports PC-Mall
                Estimated Quality, expected value 30.1, distribution : (0.03 0.0)(0.22 8.5)(0.08 30.0)(0.67 38.5)
                Estimated Cost, expected value 2.0, distribution : (1.00 2.0)
                Estimated Duration, expected value 6.0, distribution : (0.09 5.0)(0.09 5.5)(0.72 6.0)(0.01 7.0)(0.01 7.5)(0.08 8.0)
```

Figure 4.8: Alternative Set for *Build-PC-Objects*

```
Schedule SCHEDULE_meta_root_sum_2, Utility 0.69
-------------------------------------------
| PC-Connection | PC-Mall | ZDnet-Reviews |
-------------------------------------------
Quality distribution (sum of TGs): (0.02 0.0)(0.02 8.5)(0.01 10.0)(0.07 18.5)(0.16 28.5)(0.07 30.0)(0.65 38.5)
Expected value: 33.29, Probability q or greater: 0.65, Schedule value: 33.29
Cost distribution (sum of methods costs): (1.00 0.0)
Expected value: 0.00, Probability c or lower: 1.00, Schedule value: 0.00
Finish time distribution (finish time of last method): (0.02 6.0)(0.16 7.0)(0.03 7.5)(0.28 8.0)(0.00 8.4)(0.04 8.5)(0.31 9.0)(0.17 10.0)
Expected value: 8.45, Probability d or lower: 0.49, Schedule value: 8.45

Schedule SCHEDULE_meta_root_sum_0, Utility 0.67
---------------------------------
| PC-Connection | ZDnet-Reviews |
---------------------------------
Quality distribution (sum of TGs): (0.20 0.0)(0.08 10.0)(0.72 30.0)
Expected value: 22.40, Probability q or greater: 0.72, Schedule value: 22.40
Cost distribution (sum of methods costs): (1.00 0.0)
Expected value: 0.00, Probability c or lower: 1.00, Schedule value: 0.00
Finish time distribution (finish time of last method): (0.15 4.0)(0.30 5.0)(0.35 6.0)(0.20 7.0)
Expected value: 5.60, Probability d or lower: 0.45, Schedule value: 5.60

Schedule SCHEDULE_meta_root_sum_1, Utility 0.65
---------------------------
| PC-Mall | ZDnet-Reviews |
---------------------------
Quality distribution (sum of TGs): (0.10 0.0)(0.09 8.5)(0.81 28.5)
Expected value: 23.85, Probability q or greater: 0.81, Schedule value: 23.85
Cost distribution (sum of methods costs): (1.00 0.0)
Expected value: 0.00, Probability c or lower: 1.00, Schedule value: 0.00
Finish time distribution (finish time of last method): (0.03 5.0)(0.03 5.5)(0.27 6.0)(0.03 6.5)(0.28 7.0)(0.04 7.5)(0.32 8.0)
Expected value: 6.95, Probability d or lower: 0.36, Schedule value: 6.95

Schedule SCHEDULE_meta_root_sum_8, Utility 0.30
---------------------------------------------------------------
| PC-Connection | PC-Mall | ZDnet-Reviews | Consumers-Reports |
---------------------------------------------------------------
Quality distribution (sum of TGs): (0.02 0.0)(0.00 8.5)(0.00 10.0)(0.02 18.5)(0.04 28.5)(0.02 30.0)(0.18 38.5)(0.01 40.0)(0.05 48.5)(0.12 58.5)(0.05 60.0)(0.49 68.5)
Expected value: 55.34, Probability q or greater: 0.66, Schedule value: 55.34
Cost distribution (sum of methods costs): (1.00 2.0)
Expected value: 2.00, Probability c or lower: 1.00, Schedule value: 2.00
Finish time distribution (finish time of last method): (0.02 9.0)(0.16 10.0)(0.26 11.0)(0.03 11.5)(0.00 11.7)(0.30 12.0)(0.20 13.0)(0.03 15.0)
Expected value: 11.65, Probability d or lower: 0.47, Schedule value: 11.65

Schedule SCHEDULE_meta_root_sum_5, Utility 0.26
-----------------------------------------------
| PC-Connection | PC-Mall | Consumers-Reports |
-----------------------------------------------
Quality distribution (sum of TGs): (0.02 0.0)(0.05 8.5)(0.02 10.0)(0.18 18.5)(0.14 38.5)(0.06 40.0)(0.54 48.5)
Cost distribution (sum of methods costs): (1.00 2.0)
Finish time distribution (finish time of last method): (0.04 6.0)(0.04 6.5)(0.41 7.0)(0.04 7.5)(0.00 7.6)(0.37 8.0)(0.05 9.0)(0.04 10.0)

Schedule SCHEDULE_meta_root_sum_6, Utility 0.25
-----------------------------------------------------
| PC-Connection | ZDnet-Reviews | Consumers-Reports |
-----------------------------------------------------
Quality distribution (sum of TGs): (0.20 0.0)(0.02 10.0)(0.18 30.0)(0.06 40.0)(0.54 60.0)
Cost distribution (sum of methods costs): (1.00 2.0)
Finish time distribution (finish time of last method): (0.14 7.0)(0.27 8.0)(0.33 9.0)(0.21 10.0)(0.04 11.0)(0.02 12.0)

Schedule SCHEDULE_meta_root_sum_7, Utility 0.24
-----------------------------------------------
| PC-Mall | ZDnet-Reviews | Consumers-Reports |
-----------------------------------------------
Quality distribution (sum of TGs): (0.10 0.0)(0.02 8.5)(0.20 28.5)(0.07 38.5)(0.61 58.5)
Cost distribution (sum of methods costs): (1.00 2.0)
Finish time distribution (finish time of last method): (0.04 8.0)(0.27 9.0)(0.27 10.0)(0.00 10.2)(0.04 10.5)(0.32 11.0)(0.03 12.0)(0.03 13.0)

Schedule SCHEDULE_meta_root_sum_3, Utility 0.23
-------------------------------------
| PC-Connection | Consumers-Reports |
-------------------------------------
Quality distribution (sum of TGs): (0.20 0.0)(0.20 10.0)(0.60 40.0)
Cost distribution (sum of methods costs): (1.00 2.0)
Finish time distribution (finish time of last method): (0.45 4.0)(0.45 5.0)(0.05 6.0)(0.05 7.0)

Schedule SCHEDULE_meta_root_sum_4, Utility 0.21
-------------------------------
| PC-Mall | Consumers-Reports |
-------------------------------
Quality distribution (sum of TGs): (0.10 0.0)(0.22 8.5)(0.67 38.5)
Cost distribution (sum of methods costs): (1.00 2.0)
Finish time distribution (finish time of last method): (0.09 5.0)(0.09 5.5)(0.72 6.0)(0.01 7.0)(0.01 7.5)(0.08 8.0)
```

Figure 4.9: Schedules for *Build-PC-Objects*

4.3 GPGP and $GPGP^2$

Part of this thesis objective is to support new coordination mechanisms and protocols that incorporate and utilize notions of organizational context. In a conceptual sense, the decision process direction of this thesis work relates as much to the original GPGP as it does to the recent Design-to-Criteria scheduling research. Conceptually, GPGP brought non-local information to the attention of the local scheduler (then it was the Design-to-Time scheduler). However, the non-local information was a second class object to the scheduler and not evaluated in the same light as were the local candidate actions. Accordingly, the scheduler would produce a set of schedules from which GPGP would select one that best met both the local, and non-local, concerns. In a very real sense, this thesis recognizes the interplay between the local decision process and the non-local information and concerns managed by GPGP. In this new work, non-local information will be a first class object to the decision process, thus removing the need for the coordination mechanisms or coordination controller to second guess the decision process. One view of the new proposed process is that it is moving the decision functions performed by GPGP out of the coordination module and into a different locus of control that is coupled to the scheduler, in much the same way that the original GPGP was coupled with the Design-to-Time scheduler. The difference is in the clear separation of functionality, and in the types of decisions made and the information that is used in the decision making process. Relative to the Design-to-Criteria scheduler, GPGP's decision process was incredibly simple – the discussion here is intellectual and conceptual. This clear separation of concerns will simply and clarify the coordination side of the equation, leaving $GPGP^2$ highly open and easily extensible.

Returning to GPGP: GPGP is a modularized, domain independent, approach to scheduling-centric coordination. In GPGP, coordination *modulates* local control by posting constraints on an agent's local scheduler. The GPGP coordination module is responsible generating communication actions, that is communicating with other agents (via their local communication modules), and making and breaking task related *commitments* with other agents. The coordination module is comprised of several modular coordination mechanisms, subsets of which may be applied during coordination depending on the degree of coordination desired. More specifically, GPGP defines the following coordination mechanisms (for the formal details see [22]):

1. **Share Non-Local Views** - This most basic coordination mechanism handles the exchange of local views between agents and the detection of task interactions. Exchanging local views is the only way in which agents can detect and coordinate over task interactions. The mechanism exchanges information, or not, according to three different exchange policies: *exchange none*, where no information is exchanged; *exchange some*, where only part of the local view is communicated; and *exchange all*, where the entire local view is communicated. This coordination mechanism is necessary for all other coordination mechanisms – without a local view of non-local tasks and an understanding of existing task interactions there is nothing to coordinate.

2. **Communicate Results** - This coordination mechanism handles communicating the results of method execution to other agents. It is governed by three different policies: the *minimal policy* where only the results necessary to satisfy external commitments are communicated; the *task-group policy* where all the minimal results plus the final results for a task group are communicated; and the *all* policy where all results are communicated. This mechanism is meaningless without mechanism 1 above or the following mechanisms that form commitments.

3. **Avoid Redundancy** - This mechanism deals with detected redundancy by picking an agent at random to execute the redundant method in question. The agent then becomes committed to performing the action and the other agents will have non-local commitments denoting that some other agent will carry out the task at a predetermined time. Note, the type of redundancy in question here is simple duplication of work, in contrast to the redundancy of being able to generate a similar result using different methods.

4. **Handle Hard Task Relationships** - The *enables* NLE pictured in Figure 4.7 denotes a hard task relationship.

The results from the *Get-Basic-Product-Information* task are required for the *Gather-Reviews* task. This coordination mechanism deals with such hard, non-optional, task interactions by committing the predecessors of the *enables* to perform the task by a certain deadline.

5. **Handle Soft Task Relationships** - Soft task interactions, unlike hard interactions like *enables*, are optional. When employed, this coordination mechanism attempts to form commitments on the predecessors of the soft interactions to perform the methods in question before the methods that are on the receiving end of the interaction.

As mentioned above, the GPGP coordination module modulates local control by placing constraints, called *commitments*, on the local scheduler. The commitments represent either deals that GPGP has made with other agents, e.g., agreeing to perform method M by time T, or deals that GPGP is considering making with other agents. The commitments fall into three categories:

Deadline Commitment This type of commitment denotes an agreement to execute a particular method by a particular time. Thus if agent A needs the results from a method execution being performed by another agent, agent B, and they form a deadline commitment, agent A can then plan other activities based on the expectation of receiving the results from B by the deadline T.

Earliest Start Time Commitment This commitment denotes an agreement NOT to start executing a particular method prior to an agreed upon time. This type of commitment is the converse of the deadline commitment. In the two agent scenario above, this commitment could be used to denote that while agent B should execute M by time T, it should also not start executing M before time T'.

Do Commitment This commitment is weak and simply denotes a commitment to execute a particular method at some time.

Don't Commitment This commitment is weak and simply denotes a commitment not to perform a particular method (so that another agent can use a desired resource, avoid a negative task interaction, etc.)

GPGP mechanisms embody both analysis aspects of the coordination problem and coordination protocol aspects. As discussed, one aspect of this thesis entails (conceptually) moving a portion of the analysis functionality to the main agent decision process. However, analysis functions must still be performed as part of the coordination process. With our current work on $GPGP^2$, we have come to recognize the different aspects of the coordination process and to separate them architecturally. In $GPGP^2$, protocols are designed using a FSM language. The protocols are simple to design and support notions like inheritance and specialization. This interface makes $GPGP^2$ extremely flexible and amenable to change. For instance, to modify the mechanism that handles redundancy to incorporate notions of load, one need only add a few states to the protocol and supporting code that measures the load at the local agent and transmits the measurement to the other agents. In $GPGP^2$, support code is contained in a separate library and is limited only to the code pertaining to commitment formation, results sharing, and so forth. $GPGP^2$ assumes that the decision about which actions to perform, and when, is performed elsewhere.

Chapter 5

Quantified Wide-Context Decision Making and Coordination

Previous chapters introduce the topic, identify major intellectual and research contributions of the proposed research path, and place the research agenda both within the larger community and within the local research community. As Chapter 4 discusses, the thesis work proposes to significantly enhance and extend our work in multi-agent systems by developing a new local agent decision process that is responsible for evaluating and balancing the different priorities, relationships, and trade-off options that influence the agent's course of action. In this chapter we return to the topic directly, provide examples of the situations that it is intended to address, and explore some preliminary ideas about how to tackle the proposed research questions. Chapter 6 discusses highly relevant work in detail and Chapter 7 presents a work plan for achieving the thesis goals as well as sketching an evaluation plan that will help in the validation of this research.

5.1 Contextual Decision Making

Agents are socially situated which gives rise to interactions between tasks, however, being socially situated is more than interactions between tasks. Consider the fashion in which humans operate: performing a task for one's superior at work is usually more important than helping one's neighbor move a heavy object. However, if the heavy object is a life support machine, then helping the neighbor might actually take precedence. The point is that the *context* in which decisions are made matters, and that part of the context is the organization in which the agent resides and the roles it plays within the organization.

Though much of focus in the coordination research community has been on identifying and exchanging the information necessary to properly choose and sequence agent activities, recently we have come to believe, and the research community concurs [79, 53, 5, 3, 65], that the context in which the agents operate should influence their coordination activities. In our research, we have long had a view that particular behaviors are appropriate in particular situations and that other behaviors are probably appropriate in other situations; we call this concept *situation specificity*. It means literally that the actions an agent takes, and the manner in which it coordinates, are conditioned by the context in which it is currently operating. One view of this is that in particular resource-constrained situations, agents should behave differently, perhaps sacrificing solution quality in order to meet a particular deadline. This class of specialized, context dependent, targetable behaviors is embodied in our Design-to-Criteria scheduling work [87, 83, 86, 85, 84], but examples of a wide range of related work (resource-bounded control) may also be found in the larger AI community

[45, 18, 68, 95, 96]. Another example of situation specificity is to condition an agent's coordination behaviors on the environmental context. For example, in situations where the communication network is unreliable it is probably appropriate for the agents to exchange a minimum of messages and to coordinate over a limited number of (very important) task interactions. Situation specificity can also take the form of conditioning activities based on the problem instance itself. For example, in a situation where two agents have interacting task structures but the effects of these interactions are slight, it might not be worth the overhead to actually coordinate their activities. However, in a different situation, where the effects are more pronounced coordination might be worthwhile. There are a wide variety of situation specific conditionings possible; we have even done work in learning situation specific behaviors [66, 75, 74].

The work in situation specificity is akin to the community's new interest in the "larger context" because they both relate to conditioning an agent's behavior based on the context. In the research community and in our own thinking, however, the new "larger context" must include the social environment in which agents reside. This is different from our previous work in situation specificity because the idea is to condition the agents' behaviors using more information, i.e., we are expanding the scope of the conditioning. One area of new expansion in the community is regarding fully cooperative agents working toward a single common goal, e.g., destruction of a military target, as an organized team [44]. Another area of recent research takes a social view of commitments [65], agreements passed between agents to perform certain tasks (or not to perform certain tasks, as the case may be). Still another recent addition is the development of organizationally driven obligations [3, 5], where agents have power relationships between them and these constrain and limit the choices that they can make.

In this thesis, we plan to expand work in agent decision making and multi-agent coordination to address these larger, more sophisticated concepts. We feel that the process of determining a course of action, deciding what to do, with whom to coordinate, and so forth, must take in to consideration the larger context, must include valuation of the organization to which the agent belongs. At the most basic level, actions that the agent may carry out have inherent *individual* statistical characteristics, i.e., quality, cost, duration, and certainty, attributes, however, these characteristics, particularly quality (or utility), are related to the context in which they are evaluated. In other words, individual values are influenced by the larger picture, or the *organizational context* – the primary objective of this thesis research is to understand the context, model it, and create analysis procedures for working with it.

Tambe's work in agent teams is a small step in the right direction. He has come to realize that the team context is different from simple cooperative behavior. Casting his experience into our developing organizationally-centric view, a team is a persistent organization in which all members cooperate fully with one another. Tambe's recent work [53, 79] is closely related to our new direction, but his work does not address organizational concerns explicitly. In effect, notions of strong cooperative behavior are built into his system and there is no quantitative evaluation of comparing team objectives to individual or local objectives. In our work, we plan to support ranges of behaviors from entirely self interested to entirely altruistic, additionally, we plan to support notions of agents being part of multiple organizations simultaneously where agents can reason about the relative importance of activities performed for one organization or another.

Barbuceanu's recent work [3, 5] is also closely related to our new direction. However, he focuses on responsibilities or *obligations* stemming from power relationships within an organization. For example, a subordinate may be obligated to perform a particular task for a superior even if he or she is already committed elsewhere. In his work, obligations are a priority based construct – whichever obligation is more important wins. Detecting conflicts is a logical process – either there is a conflict or there is not a conflict. This differs from our view in that it doesn't support agents making local decisions about the relative importance of obligations, using his terminology, based on the context in which the decision is being made. Priorities or power-relationships are inflexible and non negotiable. His view vastly simplifies the reasoning process on the agent side, however, it is also limiting in that it does not afford agents a range of contextually dependent behaviors.

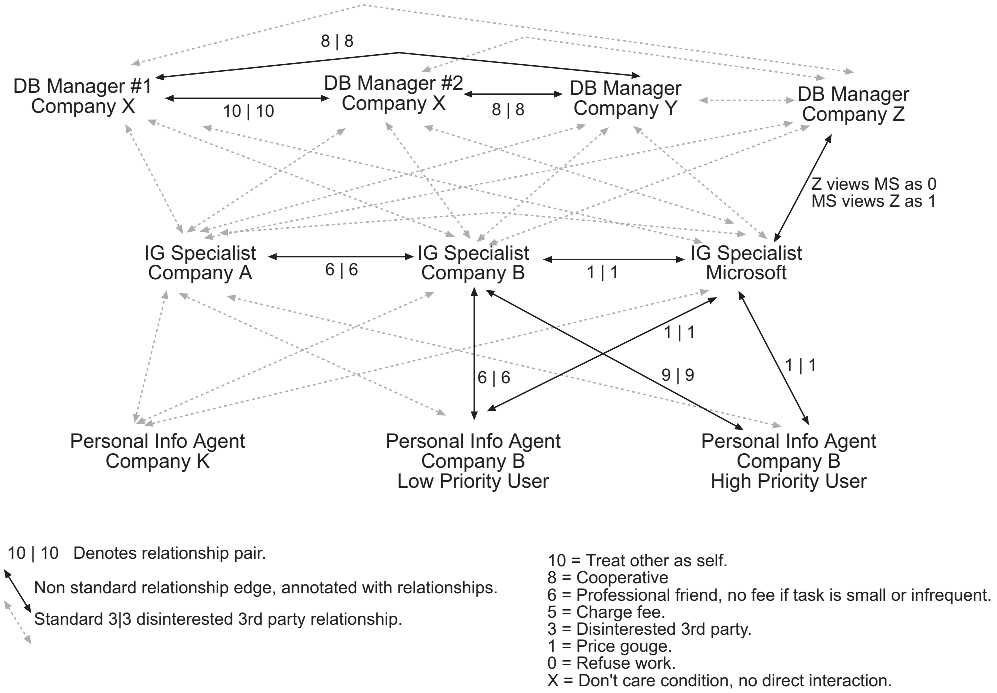


Figure 5.1: An Organized Network of Interacting Agents

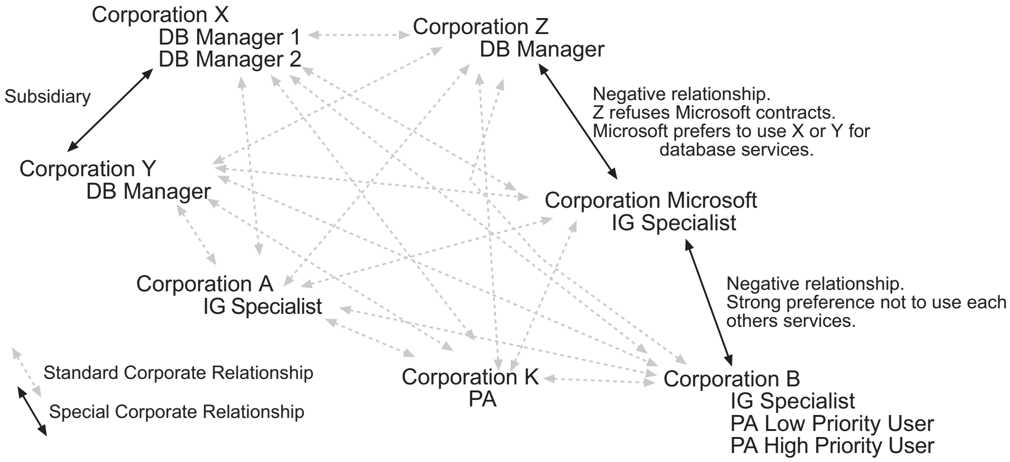


Figure 5.2: Corporate Relationships

5.2 Example Scenario

To better ground the topic, consider a coarse preliminary example of the types of agent structures, and their relationships to utility, value, and decision making, that we plan to explore in this thesis. Figure 5.1 shows an organized network of interacting information agents. The agents in this network are specialized in a fashion similar to that of the CMU WARREN [19] system; there are three main agent types:

Database Managers Agents that are experts in data maintenance and organization. These agents maintain repositories of information and act as the interface between a repository or digital library and the rest of the network. The repositories may be simple databases, collections of databases, or even entail lower-level database management agents with which the primary database manager interacts. This means that the database manager's functions are not simply to query a single existing database. It too conforms to the properties of agency, having multiple goals, multiple ways to achieve the goals, and so forth.

Information Gathering Specialists These agents are experts in particular domains. For example, one specialist might be an expert on automobiles whereas another might be an expert on software products or weather prediction. These experts know about databases (and database managers) pertaining to their area of expertise, or know how to locate such databases. Their task is to gather information, assimilate it, and produce a report, possibly accompanied by a recommendation to the client about a particular action to take based on the gathered information. These agents receive high level queries or requests for information and in response plan about which sites to query/search and handle the assimilation of the gathered data.

Personal Agents PAs are agents that interface directly with the human client, perhaps modeling the client's needs. These agents also decide with which information specialists to interact to solve a client's information need. PAs for a given company may interact with specialists outside of the company, however, interaction styles may differ.

Interactions between the agents are generally driven by a client side information need. Personal agents send requests to information gathering specialists who send requests to database manager agents. However, all agents may interact with information specialist agents at will, and vice versa. This enables the specialists to perform periodic tasks for the personal agents and to "watch" for information that might be of interest to the personal agents (where the watching may be handled by the database manager agents as well). Database managers can also interact directly with one another since it may be that an information request sent to one manager might be better (more cheaply, faster, more completely) filled by another, or that the managers have overlapping tasks and can thus share results and save time or conserve other resources. Information specialist agents can also interact directly for similar reasons.

The agents have different types of relationships depending on factors such as the relationship between their respective corporate entities, e.g., company Y is a subsidiary of company X, and the relationship between the two agents themselves, e.g., information gathering specialists A and B have a good professional relationship. The peer to peer mapping is shown in Figure 5.3 and a corporate-based organizational view is shown in Figure 5.2.[1] The type of peer relationship is denoted by a pair of integers (called *relationship specifiers*) that are indices into a relationship table or function. In this case, the spectrum ranges from 0 to 10 with 0 being the most hostile or most self-interested and 10 being the most altruistic. Specifically, level 0 relationship specifiers denote that the agent who views the other agent at the 0 level will not (knowingly) perform any work for that agent at any price. This is illustrated by the relationship between the Microsoft information gathering specialist and the database manager for company Z. The manager for company Z is hostile toward the Microsoft specialist, but, conversely, the Microsoft specialist does not regard the company Z's manager with the same degree of dislike (it regards the manager with a level 2 relationship specifier). Most of the entities have a level 3 relationship specifiers, which denotes the standard *disinterested 3rd party* behavior

[1] Other views could also be constructed, e.g., according to professional organization (IG Specialist, DB Manager, PA) or by friendships, e.g., the specialist for company A and the specialist for company B.

	DBM Co. X	DBM #2 Co. X	DBM Co. Y	DBM Co. Z	IG Co. A	IG Co. B	IG Co. Microsoft	PA Co. K	PA Co. B Low P	PA Co. B High P
DBM Co. X		10 I 10	**8 I 8**	3 I 3	3 I 3	3 I 3	3 I 3	X	X	X
DBM #2 Co. X	10 I 10		**8 I 8**	3 I 3	3 I 3	3 I 3	3 I 3	X	X	X
DBM Co. Y	**8 I 8**	**8 I 8**		3 I 3	3 I 3	3 I 3	3 I 3	X	X	X
DBM Co. Z	3 I 3	3 I 3	3 I 3		3 I 3	3 I 3	**2 I 0**	X	X	X
IG Co. A	3 I 3	3 I 3	3 I 3	3 I 3		**6 I 6**	**3 I 3**	3 I 3	3 I 3	3 I 3
IG Co. B	3 I 3	3 I 3	3 I 3	3 I 3	**6 I 6**		**1 I 1**	3 I 3	**6 I 6**	**9 I 9**
IG Co. Microsoft	3 I 3	3 I 3	3 I 3	**2 I 0**	**3 I 3**	**1 I 1**		3 I 3	1 I 1	1 I 1
PA Co. K	X	X	X	X	3 I 3	3 I 3	3 I 3		3 I 3	X
PA Co. B - Low P	X	X	X	X	3 I 3	**6 I 6**	1 I 1	X		X
PA Co. B - High P	X	X	X	X	3 I 3	**9 I 9**	1 I 1	X	X	

Key

X I Y denotes relationship where X is the view the row member has of the column member and Y is the view the column member has of the row member.

10 = Treat other as self.
8 = Cooperative.
6 = Professional friend, no fee if task is small or infrequent.
5 = Charge fee.
3 = Disinterested 3rd party.
1 = Price gouge.
0 = Refuse work.
X = Don't care condition, no direct interaction.

Notable Relationships

Co. Y is a subsidiary of X.
IG Co B and IG Co. A have a friendly professional relationship.
Co. Z is a hostile competitor with Microsoft (mostly unknown to MS).
IG of Co. A and B and Microsoft have a mutual dislike.
In Co. B, some users have higher priority than others.

Figure 5.3: Agent Relationships on a Peer to Peer Basis

where standard fees are charged for services. The same-corporate-entity relationship between the two managers for company X is modeled by a level 10 relationship specifier – neither charges the other for work and the activities of the other agent are regarded with the same preference as local activities. The subsidiary relationship between the database managers for company Y and company X is modeled by the 8 level relationship specifier – no fees are charged, and the agents cooperate, though the subsidiary tasks are not regarded at the same level as the local (main corporate) tasks and vice versa. The professional working relationship between the information specialists of companies A and B is denoted by a level 6 relationship specifier, which means that no fee is charged as long as it is convenient to perform the task or provide the result and the request is not too frequent. A level 6 relationship specifier also appears between the personal agent of a low priority user at company B and the information specialist for company B (this should probably be a level 7 specifier) – the intent is to denote that the lower priority user's requests are viewed differently than those belonging to a high level user (whose requests are entered at level 8 instead).[2]

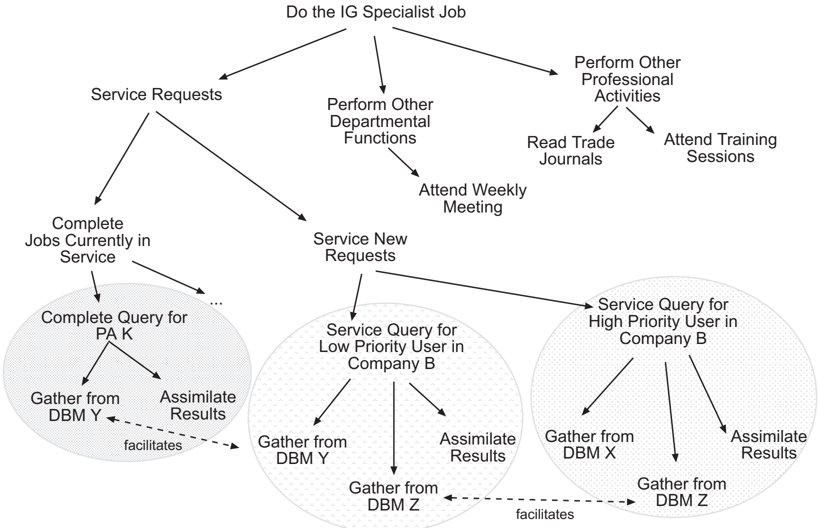


Figure 5.4: Task Graph for IG Specialist B

The purpose of the relationship specifiers is to denote the type of interaction between the two agents (ranging from self-interested to altruistic) and to denote the importance of values associated with actions carried out in this context. Figure 5.4 shows a task graph for the information specialist agent that works for company B. The leaf nodes are primitive actions and all subtasks are joined under an *or* function, meaning any combination of the subtasks may be performed and in any order. It appears as though this particular task could be cast in the TÆMS task modeling language as the structure has been specified in such a way that quality from all of the actions relates to the agent's overall objective, which is to do its job. However, as we will see, this is not actually the case.

Henceforth we will use IG_α to denote the information gathering specialist for company α, DBM_β to denote the database manager for company β, and PA_γ to denote a personal agent from company γ. Since PA_B is ambiguous, PA_{BH} will denote the high priority user and PA_{BL} will denote the agent for the low priority user.

[2] The issue of priority may relate to different marginal costs associated with the different agents. However, rather than regarding higher priority as being correlated with a lower marginal cost, it is perhaps more appropriate to view higher priority as being associated with a higher marginal utility, while the costs remain constant. In this case marginal utility is accumulated both by the consumer agent, as in standard economic models, but also by the producer as it gains some intangible value for working within its organization. Returning to the issue of marginal cost, it could be that the intangible value in this case is some form of revenue, and that the higher priority user "pays" more of it for the services, and thus has a lower marginal cost even though the cost to produce the services are the same for both the high and low priority clients.

As shown in the task graph, IG_B has many different alternative tasks from which to choose its course of action. The problem now becomes making the choice in such a way that the output reflects the different relationships this agent has with the other agents. Given the relationship specifiers between IG_B and PA_{BH} / PA_{BL}, it is not difficult to estimate that doing tasks to service the request from the high priority PA_BH should have a higher associated value than doing tasks for the low priority PA_BL. However, how do we relate these requests to the request stemming from PA_K? The relationship between IG_B and PA_K is a $3|3$, however, all this says is that PA_K should pay for the service. It does not relate the value of performing tasks for PA_K to the value of performing tasks for the PA agents from company B. This points to a problem with the way even this simple example is framed. Clearly there is a difference between the *agent interaction style* and the way in which action *value* is determined. In this case, IG_B agent has a *self interested* style with PA_K (it charges a fee), while it has a *cooperative or altruistic* style with the PAs from its own company. However, how do we relate the value stemming from financial rewards to the value stemming from doing intercompany tasks? It appears as though we need a relationship specifier pair, or triple, to model the way in which one agent regards another and evaluates actions stemming from that relationship. Items that may be part of the equation include:

- Interaction Style – self-interested or cooperative. When cooperative, no money changes hands, and value originates from some other (yet to be determined) source.
- Valuation – specifies how are values computed given the interaction style. If all exchanges were cast in a monetary framework, we could associate high-value with net profit. Agents could then be motivated purely by profit.
- Negotiation Style – agents might negotiate differently depending on their relationship. This appears to be independent from the interaction style issue.

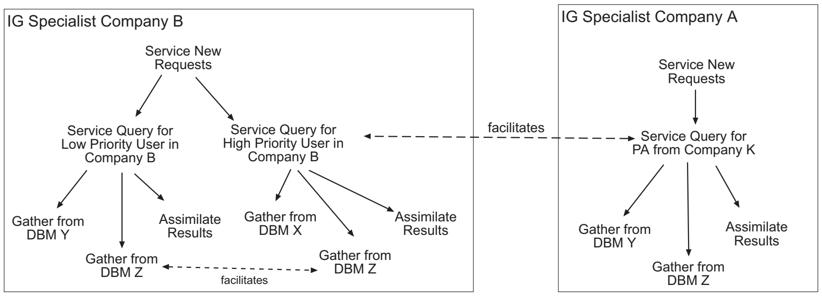


Figure 5.5: Task Graph for IG Specialists B and A

This simple high-level scenario illustrates one aspect of the types of issue this thesis will address, namely, relating activities that stem from different sources in some way as to maximize the value to the local agent, and possibly to the larger organization as a whole. However, this first situation is somewhat misleading in its simplicity because the example is strictly a task allocation or contracting situation – the issue is wholly one of relating local value between *domain actions*. There are not any communication or coordination actions to muddy the situation. Consider an expansion of the scenario. Figure 5.5 shows the task graphs belonging to IG_B and IG_A. In this example, the agents have candidate information requests that overlap and the results of one agent's search can greatly improve the other agent's results by lowering its expected duration and its expected cost. At this point, the agents negotiate to see who is going to do the work and who is going to receive the benefit. Given their identical relationship specifiers, namely $6|6$, one will volunteer to do the task for the other without any compensation. (This is determined through a coordination protocol that may be simple or extremely complex. However, the protocol must rely on decision-level support for the expression and evaluation of value stemming from the joint activity.) Assume IG_A volunteers, this results in the situation shown in Figure 5.6. The problem is now in valuing the activities within IG_A. It seems as though the value from its original actions (e.g., *Gather-from-DBM-Y, Gather-from-DBM-Z*) should increase, because it has now agreed to provide the results to IG_B. However, there is another issue. Because it is going to share its results, it now must also factor in the cost of packing the results and sending them to the other agent. Where does the value of this action come from? And where is it accrued or propagated within IG_A?

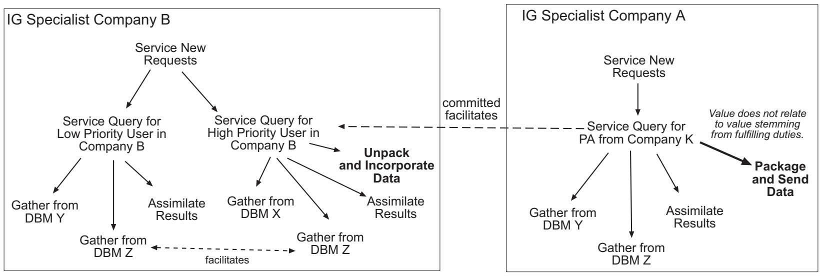


Figure 5.6: Task Graph for IG Specialists B and A

This scenario does not provide many answers, its purpose is to illustrate the types of situations that this thesis will address. The primary issues are: 1) modeling organizational and inter-agent relationships and knowledge, 2) relating the knowledge to the valuation of tasks and actions, 3) reasoning about the valuation of tasks from this new perspective to decide which actions to take, and when. Enhancing the decision process at each local agent to include the larger context will facilitate the use of existing coordination mechanisms (e.g., GPGP) in larger multi-agent systems and it will support new coordination mechanisms and protocols that explicitly leverage the new contextual information. Decision level support and explicit organizational structures will enable coordination protocols to determine dynamically with whom to communicate and coordinate, and the degree of effort with which to pursue joint action with any given agent, set of agents, or organization of agents.

5.3 The Topic

Our work will significantly extend multi-agent coordination research by focusing on advancement and expansion in two primary areas:

Area 1: Information and Structure We will expand and restructure the types of information that agents build, maintain, exchange, and reason from. An important requirement of all aspects of the information side of this thesis work is the continuance of the TÆMS tradition of *domain independence*. The main aspects of this extension and restructuring include:

- **Organizational Knowledge** We will explicitly model the organizations to which an agent belongs and the roles associated with the organization. Included in this knowledge are notions like how an agent should coordinate with other agents in the organization, perhaps specifying particular protocols or a particular medium of exchange, as well as the class of activities that the agent generally performs under the auspices of a particular organization. This area of thesis work has two main requirements: 1) understanding what information to model, and 2) structuring the information so that the agent can reason from it. The addition of explicit organizational models will facilitate experimentation with agents that coordinate in a self-interested fashion and in an altruistic fashion. It will also enable us to build agent systems in which the agents may form temporary relationships with other agents and/or to collaborate in multiple different organizations concurrently. This knowledge extension will also spur work in coordination protocols for forming and maintaining organizations – an activity that is inherently different than the activity of simply temporally sequencing actions and coordinating over resources.

- **Explicit Joint Goals and Joint Intentions** In GPGP, $GPGP^2$, DTC, and TÆMS, [25, 27, 29, 22, 88, 61, 59, 23, 26, 87, 83, 86, 85, 84], NLEs or task interactions model instances where the problem solving activities of agents may overlap. This is an implicit representation of what is formally known in the multi-agent systems literature as a joint goal or a joint intention [50, 40, 11, 78]. The implicit representation, while generally usable, makes it difficult to separate the motivation for coordination from the exact mechanics of the effect. In other words, while we model a hard task interaction as an enables NLE between two agents, we lack a separable notion of why the tasks interact – they interact because the agents share some common goal or subgoal. We have discovered a few different classes of problems where our interaction modeling does not properly motivate coordination (details below), i.e., GPGP fails to coordinate over particular classes of interactions. One possible solution is to change the modeling or to add new nles that describe the relationship. However, it is our belief that moving to an explicit representation of the joint goals that underly the interactions, and motivate coordination, is the more principled approach to coping with the problem. With explicit joint goals the desire to coordinate can be driven by the existence of joint goals rather than the description of nles. This may also lead to future work in coordination protocols that deal with joint goals, and the recognition and formation of these, in addition to our current protocols that deal with results sharing and temporal sequencing of activities.
- **Commitment and Decommitment** Currently the value or utility associated with satisfying a commitment given to another agent is not regarded as a first class object in an agent's decision process. This is because the decision process, which is embodied by our DTC scheduler, is focused on the quality, cost, and duration characteristics of the domain problem solving actions – not the value associated with cooperating with other agents. This myopic view is natural given the evolution of our work; our work has evolved from from application-specific coordination in distributed interpretation problems to generic methods for agent control and coordination. Though natural, the view is lacking. The relative importance of all agent activities should be evaluated from a unified perspective. The value of satisfying a commitment given to another agent, or conversely the penalty for breaking a commitment, should be compared to value associated with all the other actions available to the agent, including domain problem solving actions. This is common sense when looking at socially situated human problem solvers – providing verbiage for one's coresearcher's paper has value even if the task is not central to one's primary research direction. We will explore this issue in greater detail later.
- **Support for Default Knowledge** Default knowledge is one type if situation specific conditioning. In certain circumstances, agents might know in advance how they are to interact with other agents, which actions to perform, etc. In much of the community, agents communicate in order to coordinate. However, some research [47] centers on agents inferring the plans of other agents and thus deciding what to do without explicit communication. A related idea is learning which actions to take off line, a priori, and then using the knowledge during subsequent problem solving. While this thesis will not focus on the role of default knowledge in coordination, we will make explicit efforts to design the rest of the information structures so they are amenable to future extensions in this area.
- **Support for Behavioral Axioms** Whereas "default knowledge" connote replacing information that is generated by problem solving or coordination with pre-stored knowledge, "behavioral axioms" define conventions that agents must follow when interacting with other agents. The notion of formal behavioral axioms has become increasingly important in MAS as formal ideas of intent and cooperation are explored.

The boundary between behavioral axioms and default knowledge blurs if one considers the possibility of situation specific axioms, e.g., in a hostile situation decommitment does not entail communication, whereas in other situations it does. This thesis will not explore behavioral axioms per se, but, as with default knowledge, the information structures and decision process will be designed in such a way as to facilitate future work in this area.

- **Knowledge Structure** TÆMS is a framework for modeling domain problem solving activities; quality propagation in TÆMS describes how primitive actions contribute to achieving some overall objective. It is difficult to relate this notion of domain quality or performing a domain task to tasks that are not directly related to the domain, e.g., satisfaction of commitments made to other agents or performing a task related to some organizational obligation. Aside from the issue of what-one-is-modeling, there is also an issue of combinatorics when reasoning with a TÆMS structure. As we discuss in Section 4.2, the general problem is exponential. In this thesis, we will examine the possibility of constructing a different, possibly more abstract and efficient, framework that structures and relates commitment value, communication action value, and domain problem solving value. Adding new quality-accumulation-functions to TÆMS that enables the scheduler to reason about commitment value and decommitment cost from the same, very detailed, perspective that is used for evaluation of primitive actions is another possibility.

Area 2: Wide-spectrum Contextual Decision Making As discussed, in our current work and in the work in our field, agents make decisions about which actions to perform, when, and for whom, typically using very focused knowledge about the actions themselves. We will incorporate the organizationally-centered models (above) into the agent decision process so that agents consider a much broader context than domain actions. One of the main requirements of this aspect of our work is that the new decision process use a resource-bounded approach to reasoning (the algorithms must operate in soft real-time). At this time, it is not clear whether the enhanced decision process will take place within the current Design-to-Criteria scheduler or whether a new framework will be used (possibly in conjunction with the DTC scheduler). This determination is part of the thesis work.

5.4 Preliminary Sketches

In this section we explore preliminary ideas about the types of information to be stored and organized in the new knowledge structures and how this information might be related and reasoned about.

5.4.1 Organizational Knowledge and Roles

In this new work, agents will have organizational roles and a given agent may have multiple organizational roles. That is, it may belong to multiple organizations (groups of agents) simultaneously, and within each organization it might have a different role. What exactly we mean by organizational role is a research question, however, a possible data-structure-like view of some organizational roles appears in Figure 5.10. In English, roles:

- Partially define the duties that the agent is likely to perform within the organization.
- Partially specify the other agents belonging to the organization.
- Possibly contain abstractions or models of the problem solving behaviors of other agents in the organization, i.e., contain information describing likely activities of the other agents. This type of organizational knowledge is closely related to the organizational roles found in Durfee's work [32], as discussed in Section 6.1.4 (which are essentially abstractions of the actions that an agent is likely to carry out).

- Specify the relationship between agents within the organization and between member agents and non member agents.

- Specify the relative importance of tasks carried out to address the needs of the organization. In other words, the role defines how important tasks being performed to meet the needs of the organization are to the agent. This enables the agent to weigh the importance of tasks being performed for a given organization against those being performed for another organization.

- The organizational role may also specify coordination protocols that will be used to interact with other agents in the organization. This includes notions of self-interested or fully cooperative behavior.

- The roles may also define a priori joint goals or commitments, i.e., things that are somewhat static and typical for the organization. In other words, agents may always perform certain activities with other agents and these can be specified in the organization role rather than discovered anew each time by the agents.

- The importance or value of satisfying particular organizational roles is also contextually dependent. That is, agents must reason about their different roles and the importance given to a particular role is relative to the other roles in which the agent is involved.[3]

- Organizational roles are also dynamic. They can change as the result of on-line learning or by direct communication with other agents. For example, an agent may form an organization during problem solving and create a new organizational role for that instance. Note: This aspect of a dynamic organization implies that there are a set of axioms or rules governing organization formulation.

- Roles may relate to the ability of agents to form groups that are regarded as outside agents as a single agent, i.e., the collective bargaining or collective coordination model. Roles address this model by providing a means to characterize likely activities that an agent may perform as well as power relationships with other agents in the organization. One can envision a duly elected leader building a role description for the entire group and then negotiating with other group leaders from this abstract view. The actual implementation of this model is beyond the scope of this thesis work – the relationship is that this research will provide the necessary tools on which such a model may be constructed.

Relating this conceptualization of an organizational role back to the network of information agents, the agents in the network often have more than a single role, and the roles vary in terms of their specificity. Consider IG_B, it belongs to the organization of all agents working for company B, it also belongs to the organization of information gathering specialists, it also has a professional relationship with IG_A, which again entails a different organization. Clearly, precedence relationships or specificity rules must be defined. The relationship with IG_A perhaps modulates the the preferences expressed in the corporate organization, though one can envision scenarios in which either organization should dominate the other. This bears further examination.

5.4.2 Joint Goals

This organizational knowledge forms the top-level of the agent's view of its activities. In some sense, organizational knowledge is the most general, and the most broad. Organizational roles are related to, or grounded in, specific goals that the agent may pursue. In other words, agents choose to work on goals, and the context in which the goals are evaluated and weighed against alternative goals is provided mostly by the organizational role associated with the goal. Goals are characterized as follows:

[3] Organizational roles, goals, commitments, and primitive actions are all intertwined in a bidirectional influence web.

- Goals are always associated with an organizational role. Note, if the goals are "individual" goals, then they are associated with a very small organization, i.e., self.
- Goals can be either pending or active. Active goals are those that are currently scheduled (the agent has an intention to perform some work relating to the goal) or those that are being explored in order to form an intention. Pending goals are candidate goals that the agent has not decided to work on yet, but, that may be selected for work in the future.
- Goals are characterized statistically, i.e., goals have some utility associated with their pursuit.
- The characterization of the organizational role to which the goal is associated *affects* the characterization of the goal. The details of this must obviously be worked out, however, the important notion is that goals have value or utility (and characterizations) on their own, but, that the view that the agent takes of this utility is defined by the organizational role. For example, a goal that is very important for one organization might be less important than a less important goal that relates to a different organization to which the agent is much more strongly related. In human terms, a moderately important goal of one's superior at work probably takes precedence over an important goal that is being suggested by a casual acquaintance.
- The characterization of the goal may also, in turn, affect the the characterization of the organizational role. The details require further refinement, but, one can envision a situation in which the roles specify a preference for goals that are unattainable. In this case, one possible (rational) response is to change the weight or value associated with the organizational role (from which the goal stems) to reflect that previous attempts to address the role have proven unproductive and wasted effort.
- Goals come into being in a variety of different ways: 1) as the result of subgoaling on an existing goal, 2) being specified by the organizational role, 3) generated by an axiom that applies in the current problem solving context, 4) or via direct communication with other agents (where said communication and decisions based on that communication is governed by the organizational role).

5.4.3 Commitments

Commitments are even more specific than goals[4]; they are grounded in the specific actions that the agent may carry out to satisfy a goal and are situated temporally (i.e., have a time specific component). Commitments are defined as follows:

- A commitment is an obligation that a given agent has to other agents.
- Commitments are formed as a result of deciding to work on a joint goal – a goal that is held in common with one or more other agents.
- Some commitments are communicated or generated via a message exchange, however, other commitments follow from axioms, or are specified in the organizational role. An example of a commitment generated by a message exchange would be a commitment to perform a certain task at a certain time. A commitment stemming from an axiom would be a commitment to inform the other agents working on the same joint goal if something changes to make that joint goal no longer possible. The GPGP notion of commitment is probably a subset of the commitments that can be obtained via communication. Note: commitments generated by axioms can be

[4] This is somewhat consistent with Durfee's view that goals and commitments are the same artifact. In our view, agents have goals to perform certain tasks or to bring about certain effects. The goals are general and do not have a specific *intended* (or scheduled) temporal character. In contrast, commitments are associated with actions that have been scheduled or planned, and have a specific temporal character.

changed via communications with other agents or via a more specific commitment (or axiom?) that is embedded in the organizational role defining the context under which the commitment was generated. In other words, in certain situations, different axioms may hold and commitments may be changed via communication.

- Any commitment that can be formed via communication can also be specified a priori through part of an organizational role, i.e., commitments learned or hardwired can replace communication generated commitments.

- Commitments are specific things. If they are derived from axioms, then they are instantiations of axioms. If they are generated by communication, then they are specific to the subject of that conversation. In other words, a commitment has a context, i.e., it is associated with a specific joint goal and a specific set of agents.

- In all cases, commitments relate to work that one agent is doing in conjunction with other agents.

- Commitments are not all equally as important or equally valuable. Commitments are characterized in terms of q,d,c (or other?) and their utility is constrained by the utility of the joint goal to which they are related, and the utility of the joint goal is constrained by the utility of the organizational role to which it relates.

- It is important to note that even in situations where the utility assigned to a particular commitment is static, i.e., the goal and organizational role that the commitment relates to are static, as is the commitment itself, the relative importance of the commitment to the agent is not static. I will go into this in greater detail below, but, the gist of it is that the importance of any candidate action, commitment, etc., is relative to the current problem solving context and the other candidate tasks, commitments, etc., belonging to the agent. Dynamism in commitment thus happens in two ways – commitments can be explicitly re-evaluated, producing new characterizations or new utilities, perhaps via communication with the agents involved in the commitment, but also by changes in the agent's candidate options.

- In the new view, there is a cost of decommitment associated with commitments and the cost is not directly tied to the utility of the commitment. Or maybe it is? Associating cost with decommitment is important because other agents may plan future activities based on commitments made by a particular agent and if said agent decommits at the last minute, the cost to the other involved agents may be quite high, even if the task in question was fairly trivial. (This suggests an increasing utility curve as the deadlines for time-related commitments approach, or, an increasing cost curve.) As discussed in Chapter̃efintro, associating a cost with decommitment promotes a measure of stability in the multi-agent system and is a realistic mechanism to employ in a context where the agents belong to different organizations and have different relationships (i.e., where self-interest is involved).

- Regarding partial satisfaction of commitments – one solution is to decompose commitments where appropriate. However, this really gets at the notion of commitment satisfaction not being a boolean thing but instead having degrees of goodness in terms of commitment satisfaction. Instead of viewing a commitment as a simple agreement about the time at which tasks will be performed, as is seen in GPGP, let us view commitments as 1) being characterized statistically in terms of their properties and importance, but also, 2) having associated with them utility functions that specify how the statistical characteristics of the action being performed to satisfy that commitment map into utility for the other agents involved in the commitment. Perhaps these are different sides of the same issue, but, it seems to me that the issue of deciding whether or not to satisfy a commitment (which uses the statistical characterization of the commitment) is inherently different from the issue of how well a commitment was satisfied by a particular action. For example, a given commitment might be equally well satisfied by a fair-quality result provided long before the deadline, as it is by a high-quality result provided moments before the deadline.

Figure 5.7 shows the different levels of specificity of the components discussed above. Intentions in this figure denote planned and scheduled actions. Commitments are centered around the performance of actions and the communication of results. However, commitments may also deal with the agent's behavior surrounding joint goals – what if a given goal is no longer achievable? Then the agent should evaluate whether or not it needs to communicate this information to other agents...clearly, part of the confusion with this issue also overlaps to the issue of what is contained in the axiom base. In any case, the general idea is that goals are further from the intentions level than the commitments, and that organizational roles (also viewed as STRATEGIC INTENTIONS by Haddadi) are further still.

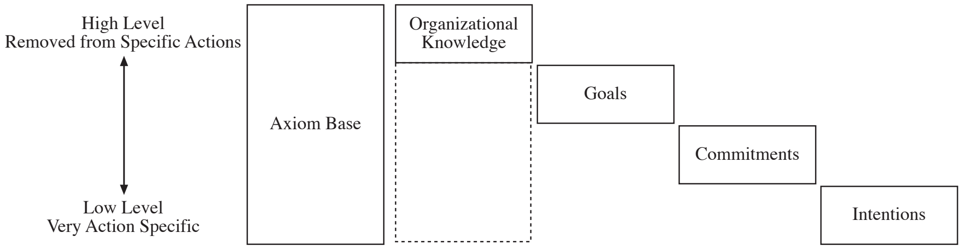


Figure 5.7: Different levels of specificity with respect to actions

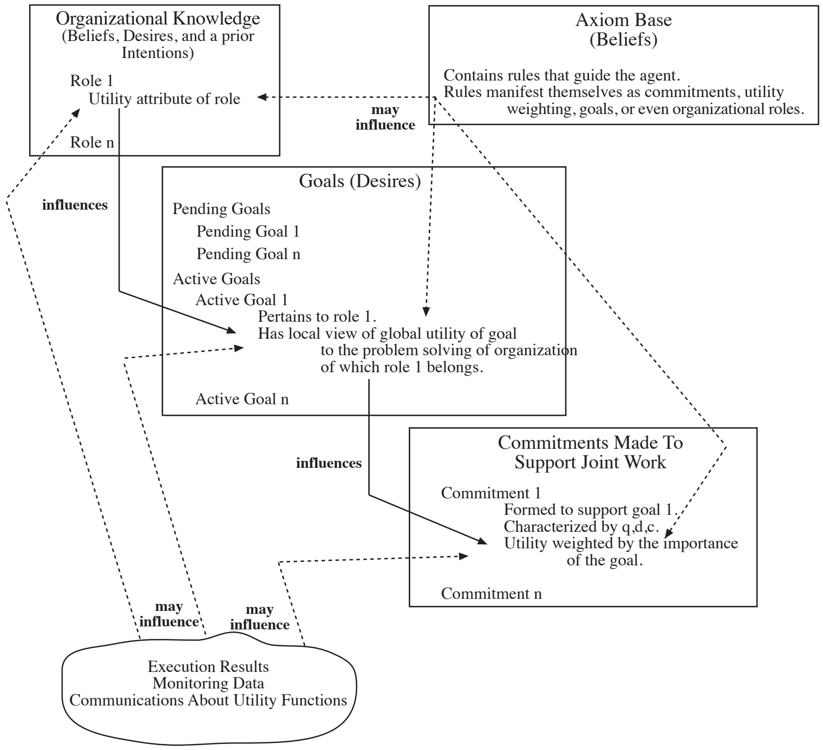

Organizational Knowledge
(Beliefs, Desires, and a prior Intentions)
Role 1
Utility attribute of role
Role n
Axiom Base
(Beliefs)
Contains rules that guide the agent.
Rules manifest themselves as commitments, utility weighting, goals, or even organizational roles.
may influence
influences
Goals (Desires)
Pending Goals
Pending Goal 1
Pending Goal n
Active Goals
Active Goal 1
Pertains to role 1.
Has local view of global utility of goal to the problem solving of organization of which role 1 belongs.
Active Goal n
influences
Commitments Made To Support Joint Work
Commitment 1
Formed to support goal 1.
Characterized by q,d,c.
Utility weighted by the importance of the goal.
Commitment n
may influence
may influence
Execution Results
Monitoring Data
Communications About Utility Functions

Figure 5.8: Flow of Quantification

5.5 Quantitative Valuation of Actions

If a given agent belongs to multiple organizations simultaneously, then it must decide what tasks to perform, and when, by reasoning about the relative importance of all possible courses of action. It is important to note that agents may be given tasks that pertain to a particular organization, but they may also have tasks of their own that do not relate to an organization (or maybe this is just an organization of one), e.g., self diagnose actions or other actions that have no direct affect on other agents. Agents must weigh the importance of local tasks (tasks not part of an organization) against organizationally-centric tasks and decide which tasks to perform, which commitments to offer to other agents, and must also reason about possibly breaking commitments already given to other agents and these types of costs.

There are three main issues in terms of the quantification of these candidate activities: 1) how to quantify value at different levels of representation (organizational, joint goal, commitment, primitive action), 2) 2) how quantifications at different levels relate (e.g., organizational versus goals versus commitments), 3) how to evaluate them. For now, let us assume that commitments, goals, primitive actions, etc., have a utility that can be computed from some statistical characterization of the item in question. Given a utility metric for each of these items, the problem is relating the metrics. Figure 5.8 sketches the flow of quantitative influence in the proposed agent structure. The primary influence flow is from the organizational level to the goal level to the commitment level. Missing from the diagram is how these relate to the selection and evaluation of primitive actions – both the utility of the goal and the utility of any commitments pertaining to the primitive actions must be considered during evaluation. This again points to the development of a new knowledge structure for representing and reasoning about these values.

From the Design-to-Criteria work, we understand what it means to consider the relative utilities of different possible primitive actions that the agent may carry out. However, in the new agent view, we must evaluate more than just primitive actions. We must be able to evaluate the cost/benefit of keeping or breaking various commitments, of pursuing a given goal, etc. The likely solution path entails developing new knowledge structures for reasoning at different levels of specificity or different granularities, making decisions on this coarse view, and then representing a subset of the candidate options in a TÆMS task structure for consideration by an enhanced Design-to-Criteria scheduler. One aspect of this integration may be the development of new quality-accumulation-functions that relate commitment value to the value of primitive actions, or that enable the scheduler to evaluate these items from a more unified perspective. The issue of how to value the different activities in which an agent is engaged, and how to relate different structures at different levels of abstraction and different degrees of specificity, is one of the primary research questions of this thesis.

5.6 Example in the Small

In this section we explore an example involving only two agents. In contrast to the previous example, the purpose here is to attempt to illustrate how notions of joint goals and commitments pertain to the process of coordination and local agent decision making. The structures presented here, and the way in which they are related, are in the earliest stages of development and may bear little resemblance to the final output of this research.

Figure 5.9 shows two agents, each having a task to "move the picnic table," albeit for different purposes. Agent A needs to move the picnic table in order to mow the lawn and Agent B needs to move it into the shade to make a better dining atmosphere.[5] The agents must work in concert to move the picnic table as neither agent can perform the task alone.[6]

[5] This distinction is important – the the agents do not have identical top-level goals or task structures and they are not operating in a strong "teamwork" type setting.

[6] This example illustrates a case that is not addressed properly in GPGP and the reason for this failure is the lack of an explicit notion of joint goal. It is important to note that the movement of the table is not analogous to the assimilation of results, i.e., we cannot fix the problem as it has been suggested by doing a better job of modeling.

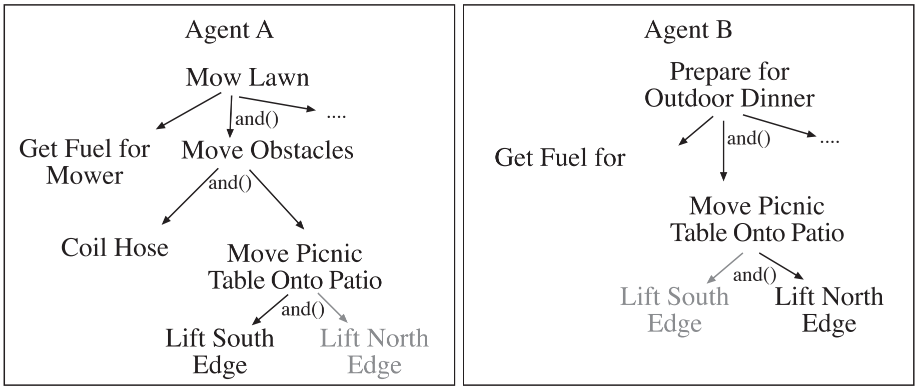


Figure 5.9: Two Agents Share a Goal That Requires Cooperation

Each agent's organizational role set is shown in Figure 5.10. The agents belong to different primary organizations; Agent A belongs to the "house care organization" and the "auto care organization," Agent B belongs to the "exterior entertainment preparation organization." However, the agents are related through a common organizational superclass, "house external organization," and it is from this commonality that their cooperation is framed. Note: if the agents did not have a common organizational role of some form, they would have to produce a new organization (bottom-up) from the task interaction. In the new context-driven view, all aspects of problem solving (commitment, joint goals, local actions, etc.) are evaluated in the context of an organization; if an organization does not exist to cover a particular interaction, a new organization must be formed before these other items can be properly understood and reasoned about.

The organizational role that covers the interaction, Role 1 for Agent A and for Agent B, determines the importance of the interaction and provides guidelines for how the interaction will be handled. Note the specification of which protocols to use in the associated organizational roles and the organizational style; the agents will interact as peers and in a cooperative fashion (these should probably be expressed as degrees, i.e., some numerical value selected from a range of possible values, as the *relationship specifiers* from the previous example)). The organization also specifies behavioral axioms – the agents should be pro-actively cooperative and offer commitments in situations such as this rather than "holding out" for the other agent to offer first. (Such axioms are not the focus of this thesis research, though we will show how they may be integrated into our work.)

The next stage is for the agents to form local views of the joint goal that identify the goal and provide an explicit context from which to reason about the goal, Figure 5.11. The joint goals are linked to the specific task and used to embody information about the task that is not specific to the task itself, but, is specific to the task *in this particular organizational context*. It is important to separate the notion of a task from a joint goal for several reasons:

1. This enables explicitly reasoning about joint goals.

2. Because the joint goal requires context beyond that which is required by the task, i.e., it is a separation of concerns just as we would find in a well designed object oriented program.

3. Because the status of the joint goal may change while the task itself remains unchanged. For example, in a situation (other than the "and()" case above), one agent may change its candidate intentions in such a way as to remove the goal from its local space, i.e., the goal may convert from being a joint goal to being a local only goal (belonging to the other agent), in which case we can just delete the associated joint goal. It may also be that the task becomes a goal in common with many other agents – separating the task from the joint goals enables us to reason about each of the goals individually, or collectively, as we choose.

Agent A
Organizational Role Set

Role 1

Organization Name: house care organization
Organizational Superclasses: house external organization, house maintenance organization, physical property mainenance organization.
Duties: mow lawn, trim shrubs, water garden, weed garden
General organization style: peer, cooperative
Protocols to use: discovery(peer, cooperative), information exchange(peer, cooperative), sequencing(peer, cooperative)
Importance of agent to organization: 50% - average
Importance of role to agent: 90% - agent's primary role
Behavioral axioms: broadcast when entering/leaving organization, offer/initiate commitments on interaction detection, broadcast when joint goal is impossible to achieve.
Agents:

Name:	Roles:	Currently In Organization?	Power relationship:
Agent B	host BBQ	yes (relationship through superclass)	peer
Agent C	exterior house maintenance	yes	peer

Role 2

Organization Name: auto care organization
Organizational Superclasses: physical property maintenance
Duties: hose off car
General organization style: hierarchical, cooperative
Protocols to use: discovery(hierarchical, cooperative), information exchange(hierarchical, cooperative), sequencing(hierarchical, cooperative)
Importance of agent to organization: 75% - above average
Importance of role to agent: 10% - secondary role
Behavioral axioms: broadcast when entering/leaving organization, offer/initiate commitments on interaction detection with superior.
Agents:

Name:	Roles:	Currently In Organization?	Power relationship:
Agent K	car wash supervisor	yes	is dominant
Agent L	sponging and scrubbing	yes	is peer

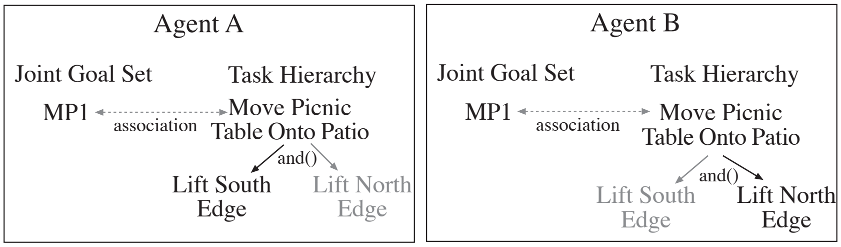
Agent B
Organizational Role Set

Role 1

Organization Name: exterior entertainment preparation organization
Organizational Superclasses: house external organization, entertainment organization.
Duties: setup non-food items, cleanup non-food items
General organization style: peer, cooperative
Protocols to use: discovery(peer, cooperative), information exchange(peer, cooperative), sequencing(peer, cooperative)
Importance of agent to organization: 90% - very important role
Importance of role to agent: 100% - agent's only role
Behavioral axioms: broadcast when entering/leaving organization, offer/initiate commitments on interaction detection, broadcast when joint goal is impossible to achieve.
Agents:

Name:	Roles:	Currently In Organization?	Power relationship:
Agent A	exterior house maintenance	yes (relation via superclass)	peer
Agent Y	food preparation	yes	peer
Agent Z	game preparation	yes	peer

Figure 5.10: The Roles of Agents A and B



Agent A - Joint Goal

Name: MP1
Joint Goal Class: and()
Relating To Organization: house care organization
Pertaining to Object: “Move Picnic Table,” type = task
Involved Agent(s): Agent B
Local Status: candidate goal
Group Status: unknown - intitalizing
Commitment Status: unknown - initializing
Local QDC Characterizations: (q,d,c distributions)
Group Utility: unknwon - initializing
Corresponding Overall Utility: to-be-computed

Agent B - Joint Goal

Name: MP1
Joint Goal Class: and()
Relating To Organization: exterior entertainment preparation organization
Pertaining to Object: “Move Picnic Table,” type = task
Involved Agent(s): Agent B
Local Status: candidate goal
Group Status: unknown - intitalizing
Commitment Status: unknown - initializing
Local QDC Characterizations: (q,d,c distributions)
Group Utility: unknwon - initializing
Corresponding Overall Utility: to-be-computed

Figure 5.11: Formation of Local-Views of the Joint Goal

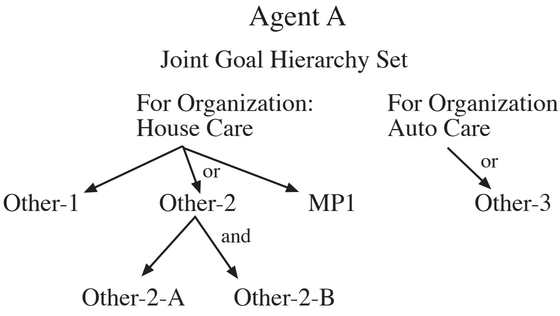


Figure 5.12: Joint Goal Hierarchies for Agent A

As joint goals are formulated, they are (possibly?) stored in a joint goal hierarchy where all the goals in a given hierarchy pertain to a particular organizational role. As goals may pertain to more than one organizational role, the goal hierarchy is graph, not a tree. The hierarchy serves as a place to reason about value associated with joint goals, in contrast to domain problem solving actions, and to understand the relationships between said goals. Figure 5.12 sketches this idea, showing the goal hierarchies for Agent A in our example. In terms of value accumulation and propagation at this level, it appears that a representation other than TÆMS is probably appropriate in order to decrease the reasoning complexity. In the figure, in lieu of TÆMS quality-accumulation-functions, the joint goals are structured using *and* and *or* functions and value is additive over the goals. The important notion that we must explore is the need for a separate mechanism to reason about joint goals and value stemming from them. The alternative is to glue the joint goals into TÆMS, however, the value that is produced by the joint goals doesn't always relate to domain value, but instead relates to the agent's overall notion of utility, which must be a function of local domain actions, and actions carried out for the non-local context (organizations, teams, other agents, etc).

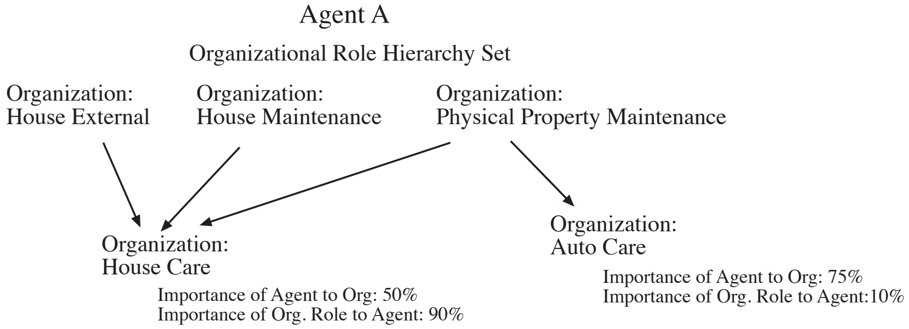

Agent A
Organizational Role Hierarchy Set
Organization: House External
Organization: House Maintenance
Organization: Physical Property Maintenance
Organization: House Care
Importance of Agent to Org: 50%
Importance of Org. Role to Agent: 90%
Organization: Auto Care
Importance of Agent to Org: 75%
Importance of Org. Role to Agent:10%

Figure 5.13: Organizational Hierarchies for Agent A

As mentioned previously, joint goals pertain to particular organizational roles and the roles affect the valuation of the joint goals. However, since agents may have multiple organizational roles, of different degrees of importance to the agent, and in which the importance of the agent may vary, value must also be reasoned about in an organizational context. Figure 5.13 shows the organizational hierarchy for Agent A. Figure 5.14 shows the linkage between the org. hierarchy and the goal hierarchy.

The idea being formulated by these figures is a very different two-way notion of utility and value. The gist of one direction of value propagation it is that activities done for a particular organization have value to the organization, and doing things for an organization has some degree of value to the agent. The other direction is that the importance of the organization to the agent affects the value associated with joint goals and ultimately tasks and commitments that stem from said joint goals. Thus, in one sense we must reason about the agent's overall utility in light of value from organizational contributions (doing joint goals, fulfilling commitments, etc.) and value from domain contributions. When choosing what action to perform next, we must examine this computation from the other direction – reasoning about and relating the values of candidate actions by somehow relating actions done for organizational value to actions done for domain value.

Some items to remember/consider as this discussion progresses:

1. The communications discussed here are taking place in the context of some coordination protocol, or a set of protocols, as specified by the associated organizational role.
2. Behavioral axioms may add candidate actions (domain or control) to the agent's set of actions to consider. An example of this would be an agent dropping a joint goal because it believes the goal can no longer be achieved. The action of dropping a joint goal may spur a behavioral axiom to produce a required communication action that notifies the other members in the organization of the agent's beliefs about said joint goal.

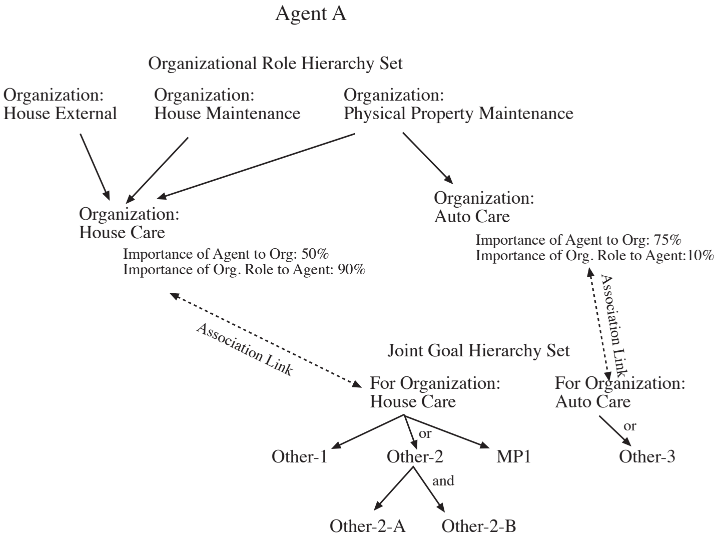

Agent A
Organizational Role Hierarchy Set
Organization: House External
Organization: House Maintenance
Organization: Physical Property Maintenance
Organization: House Care
Importance of Agent to Org: 50%
Importance of Org. Role to Agent: 90%
Organization: Auto Care
Importance of Agent to Org: 75%
Importance of Org. Role to Agent:10%
Association Link
Association Link
Joint Goal Hierarchy Set
For Organization: House Care
For Organization: Auto Care
or
Other-1
Other-2
MP1
or
Other-3
and
Other-2-A
Other-2-B

Figure 5.14: Linked Joint Goal and Organizational Hierarchies for Agent A

3. It is possible that agents will compute their utility strictly through an organizational structure. If all non-local activities are part of organizations with other agents, and all local activities are part of an organization of one (the agent), and we have a way to relate domain problem solving value to organizational value, then we can view all of these different values (joint goals, commitments, domain problem solving) from the organizational perspective.

4. The bullet immediately prior to this again suggests representing *all* primitive actions (domain, communication, coordination, control problem solving) in a task-structure-like model and reasoning about them from a unified perspective.

Getting back to the example, after the local joint goals are created and plugged into the proper structures, the agents must exchange expectations about possible execution results of their local actions, i.e. performance profiles. This step is necessary so that the other agent has some way to reason about the range of values possible for the task with which the joint goal is associated, namely *move picnic table on to patio.* [7]

For this example, let us assume that Agent A communicates to Agent B that *lift south edge* has an expected quality of (100% 10) (it can always lift the edge of the table) and an expected duration of (50% 30 seconds)(50% 40 seconds). Note, the primitive actions here are misnamed. They should really be *lift-and-move south edge* and *lift-and-move north edge* respectively to denote that the agents aren't just charged with lifting the edges and putting them back down! Assume that Agent B communicates identical expectations to Agent A about the outcome of the *lift north edge* method.

At this point, some undefined magic occurs. The agents reason about the value of joint goal MP1 and both decide to carry out the goal. Since each agent's organizational role specifies that the agent should initiate and offer commitments in these circumstances, each agent offers the other a possible *do* commitment. The agents engage in a dialogue over the commitment and settle upon some mutually agreeable time in which to move the table. At this point, commitments

[7] If, in some circumstances, values are not available or the exchange of this is not desirable, the organizational role would specify an axiom like "assume-similar-characteristics-under-a-joint-and" that would instruct the agent to simply duplicate its expectations about another method under the *and* and use these to reason from. Or, the axiom could specify default estimates or even instruct the agent to reason without estimates (probably actually requiring some unit estimate to be inserted).

are formed and characterized or valued depending on the importance of the commitment to the agent, i.e., in light of the importance of the joint goal, organizational role, and in light of the importance of the domain task *move picnic table*.

Chapter 6

Highly Relevant Research in the Field

Previous chapters discussed the topic and placed it within the research community. In this section, we explore, in detail, research in the field that relates directly to the thesis topic. In Chapter 7 we return to the pragmatic aspects of performing this thesis research and discuss work and experimentation plans.

6.1 Agents and Multi-Agent Systems

A wide range of work within the software agents community relates to this thesis direction as the thesis deals with knowledge structure, cooperation between agents, and very specialized decision theory. Some of this work was discussed briefly in Chapter 3. In this section we will concentrate on work that either most closely relates to this topic or work that is representative of a larger class of important work, to which the topic should be compared to place the thesis work within the community.

From the perspective of building organizations of cooperating agents, work within the community falls into two main groups: that which views the agents as peers with interacting actions, and that which addresses some level of structure beyond the peer level. While work in both groups is highly relevant to this thesis, it is the latter that we find most interesting and most relevant. Members of the first group include highly theoretical research in joint intentions [13, 11, 40, 50] as well as applied work like COOL [4] a language for specifying agent coordination. Members of the second group include historically important work pertaining to imposing structure on the organization of the computation [32, 15, 16] (different from organizing agents) and very recent work on broadening the context of coordination to include new notions like obligation [3, 5], social commitment [9], and rights [65].

6.1.1 GPGP and $GPGP^2$

GPGP and $GPGP^2$ are discussed in Chapters 2, 4, and 5. As discussed, the relationship between this thesis work and GPGP is mainly conceptual. In the original GPGP research, the coordination module was responsible for obtaining non-local information, forming commitments with other agents, and balancing utility obtained by local problem solving with utility obtained by coordinating with other agents, i.e., choosing the “best” schedule from a set of candidate schedules. This latter aspect of GPGP was not viewed in quite this fashion at the time, however, using the language developed in this thesis document, this is an appropriate view of the GPGP schedule-selection process.

The process, in effect, second guesses the local agent decision maker and it does this because the decision maker in this foundational work was incapable of directly reasoning about obligations made to other agents, and the costs and benefits thereof. In this thesis, we enhance and extend the local agent decision process so that it can represent and relate non-local considerations as well as simple local domain problem solving actions. This will enable us to more fully separate the coordination and decision making processes. In this work, the decision aspect of GPGP will be greatly reduced (if not eliminated) and GPGP will become primarily an expert in coordination. Given this evolution, GPGP and $GPGP^2$, and other coordination processes, e.g., contracting, will be responsible for: 1) the dialogues held by the agents, 2) the exchange of non-local information, and 3) the formation of commitments between agents. The coordination processes will rely on the new decision process to make the decision about which actions to carry out, which commitments to satisfy, and so forth, and may even rely on the new process to *focus* their activities on the formation of commitments that are deemed worthwhile based on some preliminary estimate.

6.1.2 Rights and Agreements

In [65] Jennings et al broaden the context of agent coordination to include notions of rights between agents as well as commitments stemming from actions. According to the authors, the typical view of treating agreements between agents as commitments to perform (or not) particular actions is overly specified. Agreements, the authors argue, should also include notions of obtaining *rights* to perform actions. The authors use an example of one agent wishing to query a SIG (special interest group) to illustrate the problem with agreements taking the form of commitments to perform actions. Assuming the SIG is moderated, the agent first must gain permission of the SIG moderator, and then perform its query. However, one of the conditions of the query is that the results be posted as well. Thus, in order for the agent to issue its query it must 1) obtain a commitment from the moderator that it will allow the agent to post its query, 2) post the query, but also 3) give a commitment to the moderator that the results will be shared with the rest of the SIG. This example poses a problem for the defacto approach of giving and issuing commitments. Step 1, obtaining the right to post to the SIG, does not actually entail a commitment. Neither the moderator nor the requesting agent is committed to any particular action after Step 1, i.e., the requesting agent may still *chose* to post its query or it may choose not to post its query. However, *if* the agent posts its query, then it is *obligated* to post the results to the SIG as well.

To represent situations such as these, the authors create a more general notion of an agreement between agents where the agreement can be centered on the rights of an agent to perform an action rather than a commitment to perform the particular action. In this work, commitment and joint commitment theory still comes into play in that agents must be bound to uphold the agreement in much the same way that they must be bound to uphold a commitment to perform an action (in a formal sense). The authors define a logical language for agreements using a set of propositional variables, a set of atoms expressing that an action has been performed, a set of capabilities (simply $Capable(x, a)$ where x is a agent and a is an action), a set of rights, a set of agreements, and a set of bindings of agents to agreements. The authors then define associated syntax and show how certain properties can be represented in the language, e.g., persistence of agreements and one-shot agreements. In addition to bindings of individual agents, the authors also define what it means for a group of agents to be bound to a particular agreement (they all have the agreement individually).

This work is important in that it attempts to widen the scope of issues addressable by multi-agent coordination, i.e., it moves from the conventional and very grounded view of commitments stemming from primitive actions to a more abstract concept of a *right* to perform a particular action. The process of constructing a network of rights and then commitments to perform actions can be viewed as some form of agent organization as the right, in a sense, denotes a power relationship. However, the work is not intended to address the same class of concerns as the research proposed in this thesis. For example, being *g-bound* (group bound) to an agreement means simply that each agent in some set is bound to a particular agreement (and was bound itself, not by a third party). There is no notion of agents being members of organizations outside of the context of rights and commitments – no way to describe that agent α coordinates with β from a self-interested perspective (charging money) or that β and γ are competitors and do not coordinate at all. There is also no way to frame the example from Chapter 5 in the authors' language of agreements as the example involves quantification of relationships and specifications of different interaction styles. If we convert the figure so that relationships are binary representing that an agent either interacts with another agent, or not, the structure could be represented using the agreement logic by giving only agents that are allowed to interact the right to perform communication actions with one another (and by removing the possibility of obtaining a right to interact with another agent on the fly, e.g., via delegation).

This work also approaches the issue from a logicist perspective – there is no room for quantified notions of choice. Either an agent is bound to an agreement or it is not. There is no possibility for agents reasoning about the value of upholding the agreement or the penalties for breaking the agreement. In some sense, the logical language presented in the paper could be integrated with the quantitative views of this thesis topic by providing a tool from which agents could determine whether or not they have rights to perform actions, or have agreements that they are supposed to uphold, and given this understanding the agent could then decide whether or not to uphold the agreements or whether or not it is *worth* the effort to obtain the right to perform a particular action.

It is important to realize that the thesis topic also does not deal with rights of agents, but instead focuses on the relationships between the agents where the relationships are grounded in the utility of actions. Rights of some form could be integrated into the organization specification in the form of whether or not an agent has a right to interact with another agent (whether or not it is *worth* doing) or possibly in the specification of a set of actions that the agent, as being a member of the organization, has agreed *not to perform* unless specific rights are obtained from some other agent. The issue of rights may be an interesting future extension of this thesis work.

6.1.3 Commitment, Meta-Commitment and Organizational Commitment

In [9] Castelfranchi explores coordination, joint action, and organization from the perspective of commitments between agents. The author first defines three simple types of commitment:

I-Commitment An I-Commitment is an *internal* commitment, a relation between an agent and an action – it is formed when an agent decides to perform some action at some time (selection and scheduling). The action may be for another agent or it may be motivated by a desire to achieve a local-only goal. Regardless of the motivation, the commitment is only an I-Commitment unless it is told to another agent (conveying some rights) in front of a third, witnessing agent, in which case it is promoted to an *S-Commitment*. Internal commitment corresponds to the type of commitment discussed by Cohen [11] on the basis of Bratman's work [7].

S-Commitment An S-Commitment is a *social* commitment. It too is a relational concept between at least two agents and involving a third party agent as a witness, i.e., *(S-COMM x y a z)* where x is the committed agent, y is the agent to whom x is committed, a is the action, and z is the agent. S-Commitments are more than I-Commitments that are communicated to the agent for whom the work is to be done because they transfer rights to the recipient agent. For example, y can protest if x fails to perform a after committing to do a. S-Commitments also entail an element of mutual interest. If y does not care that x has committed to do a, then x can elect to decommit from a without protest from y (this is a notion of being vested in the performance of the task).

C-Commitment A C-Commitment is a *collective* commitment, and this form of commitment is often improperly confused with S-Commitments in the MAS community. A C-Commitment is simply an I-Commitment held by many agents.

Even these simple commitment types are of interest to this thesis and to our local research agenda. Both GPGP and $GPGP^2$ coordinate through the formation of S-Commitments (though the formal reasoning about them as such is not part of GPGP). S-Commitment is also of particular interest to this thesis research because S-Commitments create expectations in the agent to whom the commitment is given, thus if the committed agent fails or decommits, it is not the simple matter of an I-Commitment where there are no consequences, i.e., Castelfranchi's analysis is consistent with the extension that there should be a cost of decommitment (reparation payment, if you will). S-Commitments are also of interest to this thesis because we are interested in exploring S-Commitments that have associated value –

in contrast to Castelfranchi's view, commitments in this thesis work will be explicitly valued via an intricate network of constructs that relate the value associated with joint goals, commitments, and organizational relationships, and the commitments will be reasoned about from this value perspective. Commitments that are not worth satisfying, will not be satisfied. The issue is valuation of commitments and an explicit decision of whether or not to satisfy them. (Related qualitative issues include determining when a commitment is *satisfied enough* or *partially satisfied*, though we will not address either in this thesis work.)

Relating these commitments to cooperative groups, the author defines a truly cooperative group as being one that is based on a common goal and mutual dependence (mutual knowledge of mutual dependence over a goal that is shared by two or more agents). Thus there is an S-Commitment between every member of the group, that is each member is committed to the group to "do its share." (This is consistent with Tambe's strong teamwork style of cooperation [78, 76, 77, 80].) The author believes, in contrast to the work done on a "social agent" by Georgeff and Rao (1992 report), that the mutual dependence is necessary in order to *motivate* work over the joint goal. [1]

In leading up to more general forms of commitment, the author gives an example of a boss agent β that knows some goal; the goal is not communicated to subordinates x and y, but, x and y are made to carry out actions to ultimately achieve the goal. Through β, x and y work cooperatively to achieve the goal, though there is no S-Commitment between the two agents. Additionally, though the agents work toward a common goal there is no identical C-Commitment held by each agent. To address this example and other, more general forms of collective cooperative activity, the author presents two new types of *organizational* commitment:

G-Commitment A generic commitment (G-Commitment) is a commitment to a *class* of actions. In a sense, the "G" status of the G-Commitment is a modifier – one can envision instances of I-Commitments that pertain to classes of actions, S-Commitments that pertain to classes of actions, and C-Commitments that pertain to classes of actions. Per the author, since "true" organizations do not spring up at the spur of the moment to handle coordination over a single action, organizations must use G-Commitments.

Generic Meta Commitments Commitments of this type are commitments to "commitment oneself to do the right thing at the right moment." According to the author, these commitments determine the structure of the organization. This is accompanied by an important concept that *the structure of the organization is different from the structures of its activities*. [2]

G-Commitments are mappable to the organizational roles envisioned in this thesis – they define a class of actions that an agent is committed to performing. However, they differ in that organizational roles as presented in this thesis may simply specify a predisposition for performing certain tasks for an organization, rather than an absolute commitment to always performing said tasks.

Generic meta commitments are somewhat analogous to the organizational relationships discussed in this thesis. Consider the case in which agents belonging to a given organization are wholly cooperative and all peers in terms of power relationships. In this situation, an agent is committed to help another agent whenever it should need help, i.e., the agent is committed to do the right thing at the right time – the relationship is akin to the G-Commitment but differs in that it pertains to relationships between *agents*, not actions. This is akin to the relationships discussed in this thesis work (i.e., in that it is about agent relationships, not just action relationships). However, generic meta commitments differ from the relationships presented in this thesis in that there is no notion of a degree of meta commitment. In this thesis we will explore degrees or ranges of different relationships, e.g., α *prefers* to work with β but it is willing to work with γ if it has sufficient resources. Castelfranchi's work on commitment is akin to the logicist view and

[1] This is *exactly* the issue with GPGP, TÆMS, and two methods under a joint and(). The involved agents must both recognize the mutual dependence of the actions, as implicitly suggested by the solution of a new NLE to model the relationship or my solution of a joint goal and recognition that methods under an and() have this dependence property.

[2] In relation to the previously mentioned Benyo, Garvey, Lesser organizational design work – the "organizational design" work is about structuring the activities only.

done without regard for quantified ranges of values. In general, Castelfranchi's commitment-based view of agent interaction is important in that it is consistent with the central concepts of this thesis, that is, creating and reasoning about relationships between agents.

6.1.4 Coordination as Distributed Search

In [32] Durfee and Montgomery define coordination as a distributed search through a space of possible, interacting behaviors of individual agents and groups of agents. A behavior is a data structure that describes an agent's activities at some level of detail and over some temporal scope. For example, a behavior might be a plan, goal, organizational role, or a schedule. Behaviors describe an agent's activities in terms of: *who, what, when, where, how* and *why*. In this work, agents coordinate by abstracting their activities to different degrees and with respect to the different attributes of the behavior data structure. For example, two agents coordinating over using a door (the work uses the producer/consumer/transporter as a problem domain) might abstract their activities and describe them in terms of where and when. Different abstractions are appropriate for coordination in different circumstances – like the action-selection-scheduling problem in this thesis, the choice of behavioral abstraction is contextually dependent. In certain cases, describing the activities in terms of who, or when, might be desirable and in other cases they might be unnecessary details. It all depends on the coordination episode at hand. Durfee views the distributed computation from the global-view discussed in Section 2.2 – the overall objective for a group of agents is to "to find a collection of behaviors that satisfactorily achieves the agents' most important goals." [3]

In this work, the authors hypothesize that "organizations, plans, and schedules have a common representation, but differ in their degree of specificity along different descriptive dimensions." The common representation in this hypothesis is the "behavior" data structure. The hypothesis is an important one, but, it oversimplifies the issues. The authors approach to this problem stems from a distributed problem solving view of MAS and is driven by the primitive actions being carried out by the agents. An organization is more than the organization of the computation – it entails relationships between the agents other than interactions between their primitive actions, e.g., power relationships between the agents. An organization also has a prescriptive role rather than just a descriptive role. Organizations can impose structure on the computation rather than being driven by the computation. Durfee's view of the organization is bottom-up and seemingly lacks a top-down component.

That being said, there is some truth to the hypothesis. Frankly, if we replace "organization" with "organizational role" the hypothesis is less disturbing as the term "organizational role" is commonly used in the literature to describe the types of actions an agent may perform. Thus from the primitive actions, to the plans used to produce them, to the organizational roles that grossly characterize the plans and primitive actions, one can imagine a common conceptual representation with differing degrees of specificity.

This work differs from the directions posed in this thesis in that it does not address how agents choose between different options and thus determine a course of action; the research focuses only on the action abstraction process. In fact, Durfee explicitly avoids the issue of choice, e.g., "it is not clear whether any generic components exist because different domains require different trade-offs with respect to criteria such as communication overhead, computation requirements, reliability, and guaranteed convergence." This statement is with respect to agent coordination but encompassed in this is the notion of choice. (Given the date of publication, we make no assumptions about whether or not these are the current views of the authors.) This work also differs in that it focuses only on the descriptive role of organizational knowledge and its view of organizations as consisting only of agents grouped by the intersections of their primitive actions. Because of this view, it also does not address the notion of an agent belonging to different organizations in any sense other than organizations as agents grouped by intersections between their behaviors. As

[3] Some where in the tech report version of this document Durfee claims that a MAS approach can decrease the computational complexity of a problem – examine his references. I think this is simply a poor word choice. I think the complexity of any problem does not change from a computational perspective, but that we may be able to solve a problem more efficiently if the solution approaches linear speedup levels.

discussed in [9] (Section 6.1.3), being a member of an organization entails more than interactions between specific tasks. If a behavior from Durfee's work could specify *who* in a prescriptive fashion, rather than bottom-up (driven by the primitive actions), it would come closer to modeling organizations as discussed by Castelfranchi. Again, because the authors focus on organizations only from the perspective of abstracting the individual problem solving activities, there is no way to describe agents having different power relationships or different interaction styles or being grouped accordingly.

Relatedly, it is not clear how notions of self-interested coordination or negotiation would be integrated into this abstraction-based coordination process. (Again because the element of choice is not described or not present.) It is also unclear, since there is no prescriptive component, how an organizational designer could construct a MAS in a top-down fashion using the behavior mechanism.

The work also ignores the interplay between the choice of primitive actions and the coordination activities – it is possible that the authors view agents as having a set of n actions that must be carried rather than n goals that may, or may not, be achieved depending on the context. In a setting where agents *decide* which actions to perform, and when to coordinate, decisions like whether or not to coordinate with another agent may change the options available to the agent in the next time step. This somewhat breaks the view of organizational roles, plans, and schedules being the same but having different degrees of specificity because the agent's organizational roles can be changed by the choice of which action to perform in the previous time step. To illustrate, say agent α decides to coordinate with β and to exchange local information so that they may coordinate. After internalizing β's information, α produces new candidate actions x, y, z and from these α's behaviors are modified. However, if α chooses not to exchange information with β, and instead performs some other action, then in the next time step α's behaviors are the same as they were in the previous time step.

Reservations aside, this work contributes to this thesis direction in its computational view of organizational roles as a specification of an agent's actions along the six dimensions (who, what, etc.). The organizational role structure developed as part of this thesis should include the six dimensions, as well as information like how the agents should coordinate and power relationships. The production of organizational roles via abstraction from primitive actions is also a useful concept. While this thesis focuses on a utility-based approach to choice, in the context of *predefined* organizational roles, future work in the creation and development of organizational roles will benefit from the behavior development process of Durfee's work, particularly the notion that the abstraction process is contextually dependent.

6.1.5 Obligations

In [3, 5], Barbuceanu describes agent coordination based on a notion of social obligations. Obligations are akin to the power-relationships described in this thesis but differ in their scope, rigidity, and lack of support for satisficing reasoned behavior. Social laws or obligations are logical constructs constrain the behavior of agents – an agent that fails to satisfy a particular obligation loses some utility. As with the approach proposed here, agents work at different levels of abstraction. Social laws determine which behaviors are more appealing to an agent (those that break fewer social laws) and the selected set of behaviors are then scheduled and planned for. On the surface, this appears to address a very similar class of concerns as this thesis work, however, the author's have greatly simplified the valuation problem in order to discuss social concerns in a multi-agent context. Some of the major differences are:

- The social model determines which behaviors are appealing and which are not, but, the obligations are not examined beyond this context. In terms of our proposed work, this is analogous to obligations identifying task groups on which an agent should work, rather than relating the obligations to utility of individual actions and then selecting which actions to perform based on a more detailed analysis. The author's approach is much simpler, but, it lacks the ability to satisfy a portion of the activities under one obligation or another. In other words, there is no *satisficing* behavior in the agent's rationality. It will either satisfy an obligation or not. There are no degrees of satisfaction and consequently little flexibility. For example, if one's boss and one's spouse have conflicting obligations, one or the other will be satisfied, but not both, even if it would be possible to satisfy some of the primitive actions for each.

- The model assumes some global generic notion of utility and it is not clear where agent's get utility, only that they pay a penalty if they do not satisfy an obligation. The issue of *from where the utility originates* is one of the questions addressed by this thesis, e.g., how do we relate the utility obtained from performing a task for a close associate to that obtained by being paid to perform some task? This question is critical in order for agents to operate in real-world environments as some tasks will be for partners, or other closely related agents, and some tasks will be related to arms-length transactions.

- Relatedly, there is no way to discuss different agents having different interaction styles based on their relationships with other agents. If agent α has an obligation to β to perform some task, and a different obligation to agent γ, there is no way to describe the situation where α and β interact via cooperative-style negotiation whereas α and γ coordinate via contracting from a self-interested perspective.

- There is no notion of commitment formation. Obligations are not formed, they exist. This is the same assumption that we will make in the initial phase of this thesis research. However, as commitments are not formed, the penalties are also not associated with decommitment. Agents reason about task performance from a very high-level perspective and do not seem to coordinate over items like temporal relationships. Either a task is performed or it is not. The work does not address the situation where some task, α, provides a result needed by another task, β, and that β must consequently wait until the result is provided before it can be executed. This may indicate that agents do not reason temporally or with respect to resource consumption about any of their primitive actions, but that they merely select the "next action" and perform it. The alternate explanation is that no coordination takes place other than the satisfaction of obligations to do or not do particular tasks.

- There is no bottom-up interaction between the scheduling of actions and the social obligations. The obligations exist. They are assigned a particular priority and higher priority obligations take precedence over lower priority ones. There is no way for an agent to contextually reason about the obligations or its alternative courses of action. For example, it is not clear that an agent would recognize if some obligation is unsatisfiable because of time limitations and thus would select another obligation to satisfy.

We have identified a portion of the differences between our work and the obligation-centered research. This bears further analysis. The gist of the matter appears to be that obligations simply define which tasks an agent will perform, regardless of context. As notions of importance or utility stemming from the obligations are not propagated into the agent's reasoning process, agents cannot make contextual decisions. The agent always tries to satisfy its highest priority obligation. It doesn't predict the possibility of failure in advance and select another obligation nor does it try to satisfy multiple conflicting obligations by satisfying them to different degrees.

6.2 Other Related Work to Examine

Chapter 7

Pragmatics

7.1 Deliverables and Research Plan

7.2 Experimental Plan

Chapter 8

Appendix A - Why a New Structure is Needed

8.1 Commitment Value is Inherently Different Than Other TÆMS Concerns

Problem: Commitment value, or the value associated with commitment satisfaction, is inherently different than the way in which we characterize the value associated with TÆMS tasks, i.e., commitment satisfaction does not map into quality.

Before exploring the rationale behind this statement, lets first discuss why it is important. In order to consider commitments first class objects, we need some way to relate the importance of commitment formation, and commitment satisfaction (or decommitment), to other local problem solving options. Obviously, the value given to commitments is influenced by the organizational role that they are associated with, however, the issue of where the value comes from is *different* than the issue of how we relate commitment value to the quality, cost, and duration characteristics associated with tasks and methods. To further clarify, with respect to commitments and valuation, it appears that there are two different issues: 1) the cost/utility benefits of engaging in the actions necessary to form commitments, and the actions necessary to communicate the data necessary to satisfy commitments (send results) or decommit (send decommitment message), and 2) the value associated with satisfying a commitment once it is formed.

8.1.1 Reasoning About Commitment Formation Actions in TÆMS

With respect to the first class of concerns, these are just a special case of TÆMS methods or primitive actions and it would seem that we can simply include them in the task structure. However, there is still an issue of relating the value of performing the actions necessary to form commitments to local (domain) problem solving actions. Where does the quality of commitment formation actions accrue? This is related to the issue of where the value generated by satisfying a commitment accrues. Figure 8.1 shows two agents with an interaction between their task structures. Agent A can facilitate Agent B's task to obtain mulch by providing said mulch, i.e., the mulch is the output of Agent A's task structure. If the agents coordinate activities, Agent B will be able to obtain the mulch for free.[1] Part of the

[1] It is possible that giving the mulch to Agent B has value for Agent A in that it means A no longer has to dispose of the mulch. This would be represented as a task to "remove mulch" in A's task structure; it is omitted to simplify the example. If said task existed in the task structure, then B's using the mulch would facilitate the mulch removal task of Agent A.

objective of treating commitments as first class objects is to enable us to reason about the cost/benefits of forming commitments versus other (domain) activities. In this example, Agent A must be motivated in order to expend energy forming the commitment with Agent B (to provide the mulch), that is we need to represent the commitment formation activity explicitly and reason about it.

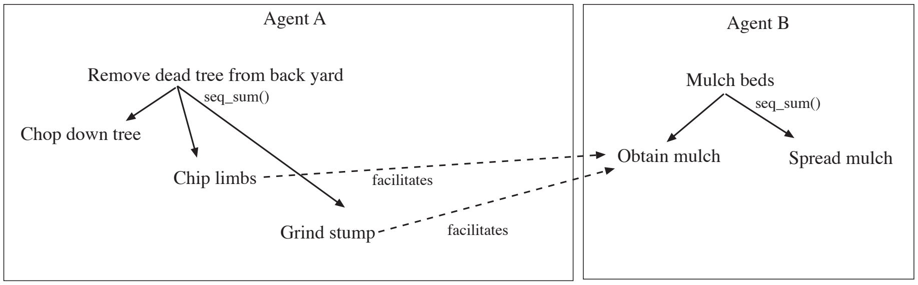


Figure 8.1: Two Agents with Task Interactions

Figure 8.2 shows one possible way to integrate the actions necessary to form commitments into the task structure. This is actually based on an algorithm developed by Wagner for integrating these concerns – but the premise of the algorithm is wrong. First, lets examine the remapping. Agent A's "Remove dead tree" task now has two subtasks, one to form the commitments and one to perform the actions. The *seq* quality-accumulation-function is used to denote that commitment formation should happen before the domain actions, but that the quality from the domain actions will propagate to the parent task. What is correct about this figure is that the quality at "Remove dead tree" is determined by the *domain actions* not the *commitment formation* actions. Since quality represents progress toward a task or goal, value obtained from commitment formation is inherently a different type of utility and should not affect the quality of the parent task. However, the use of *seq* here to "get around" the issue is in essence a kludge or a quick fix. Furthermore, what is Agent A's incentive to perform any of the commitment formulation actions? The commitment forming subtask has a *max* qaf, but since the quality doesn't accumulate or contribute to the tree removal task, it doesn't matter whether or not an action under the commitment formulation task is performed at all. The obvious modeling fix is yet another kludge – that is to attach an enablement nle from commitment formulation to the "perform domain actions" task. However, performance of the domain actions is *not dependent* upon the formation of commitments. A facilitation nle is equally inappropriate – forming a commitment has no direct value to the tree removal task thus increasing the quality, or decreasing the cost or duration of its subtasks based on commitment formation is inappropriate. All of these issues are actually the same – value and cost associated with commitment formation must be considered but it does not map directly to quality. This issue is parallel to the issue of associating value with keeping commitments that have been made or incurring penalties for breaking commitments that have been made.

Figure 8.3 shows a less intuitive mapping that attempts to illustrate one conceptual approach to the problem of differentiating commitment driven value and task quality. In this task structure, Agent A has two main tasks, to gain utility or goodwill for an organization in which it participates and to remove the dead tree from the backyard. What is correct about this figure is that it differentiates value obtained from proffering commitments from value obtained by performing the original domain actions. However, what is incorrect about this figure is the use of enablement to specify the task sequence – commitment formation does not enable the domain problem solving actions. Admittedly, we could create a new nle that is akin to enablement that denotes one activity must be performed before another (an nle based seq), however, it still is not as clean as one might wish. We need some way to combine the sequencing representation of Figure 8.2 with the differentiation of quality accumulation of Figure 8.3. In other words, the commitment formation actions relate to particular activities and they should be represented and/or sequenced with said activities. However, commitment formation activities do not contribute quality to the domain problem solving actions and we need a

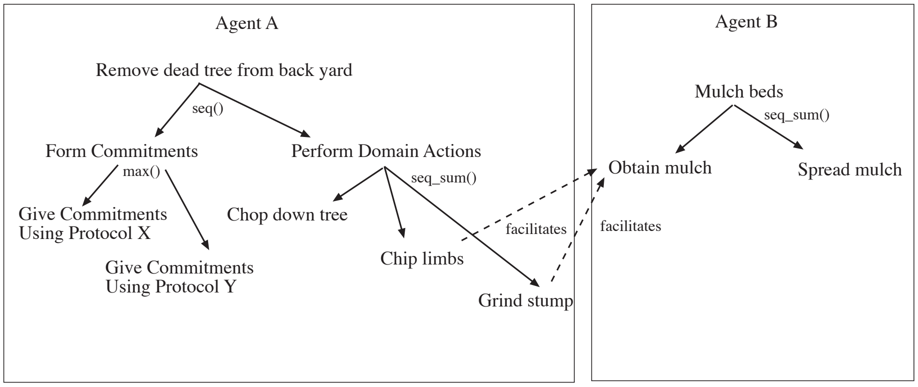


Figure 8.2: A Logical Integration that Misses the Point

different place to represent, model, and reason about value generated by the commitment formation actions.

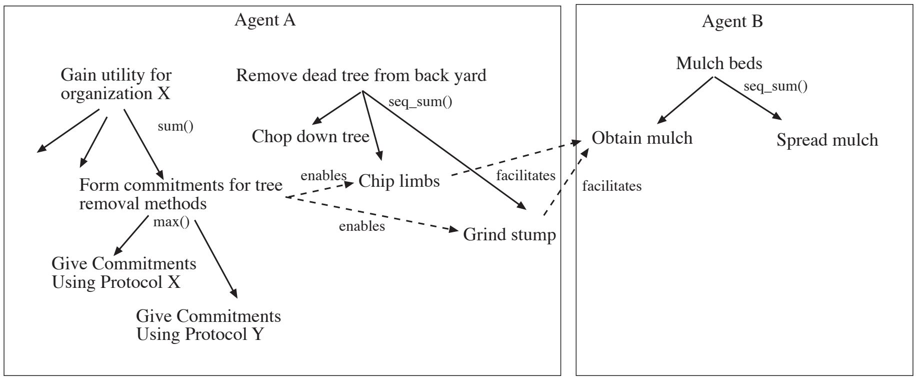

Agent A
Gain utility for organization X
sum()
Form commitments for tree removal methods
max()
Give Commitments Using Protocol X
Give Commitments Using Protocol Y
Remove dead tree from back yard
seq_sum()
Chop down tree
enables
Chip limbs
enables
Grind stump
facilitates
facilitates
Agent B
Mulch beds
seq_sum()
Obtain mulch
Spread mulch

Figure 8.3: Differentiating Organizational Value from Domain-centric Quality

One possible solution is adding a dimension to the attributes assigned to primitive actions. For example, actions could be characterized in terms of *organizational utility contributions* as well as quality, cost, and duration. The concept of a new dimension or new characterization goes hand in hand with a new objective function for the agents. The objective function would specify preferences for a given task structure, or even a given task. The idea is that in a particular organization the commitment formation from the previous example might be as valuable as the actual domain problem solving, in this case, the objective function should lead the agent to view the trade-offs between the domain actions and the commitment formation actions accordingly, i.e., in much the same light. On the other hand, in a different (possibly self-interested) organization, the agent might place little or no value on commitment formation and would thus prefer domain problem solving actions over all commitment formation actions. Note, this still does not exactly address the problem as quality accumulation and propagation have special roles in TÆMS.

8.1.2 Reasoning About the Value of Satisfying Commitments in TÆMS

Note: In this section we will use the term "commitment" to denote a commitment given by one agent to another agent, from the perspective of the agent giving or making the commitment.

After a commitment is formed, the issue is then relating the value associated with maintaining and satisfying the commitment to other options available to the agent. Even if a commitment is static, that is having static value, the importance or value of the commitment is relative to the other options available to the agent. In economic terms, in a world where there are limited resources (time, computational power, etc.), actions have *opportunity costs*, that is the performance of one action may preclude the performance of another action. The agent must be able to reason about the opportunity costs of maintaining and satisfying commitments, as well as the costs associated with decommitting, as in [1, 2]. This concept is different from the issue of what happens to the value associated with a commitment over time. For example, it might be that the value associated with a commitment increases as the commitment satisfaction time approaches, or, conversely, that the penalty for breaking a commitment increases as the commitment time approaches (as this leaves the other agent(s) less time to replan and reschedule). The issue of how to model commitment utility or value, and which models are appropriate or how many dimensions or parameters are used, is separate from the issue of how to reason about the utility and how to relate it to other candidate actions and opportunities that the agent might have. Obviously the organizational role associated with the joint goal (or shared task) and the commitment influences how the commitment is valued also, but, this too is orthogonal to the issue of how to relate the value of commitment satisfaction to the other opportunities.

One obvious approach is to propagate the value associated with a commitment to the q,d,c characteristics of the tasks and methods relating to that commitment. However, as in the previous section, this is a "quick fix" approach that blurs the issue and will make work based on commitment value even more difficult. Lets enumerate some properties of commitments that might help form hypothesis about how to treat commitments as first class objects:

1. Commitments pertain to specific tasks and methods (which are really joint goals).
2. Commitments that apply to tasks also apply to all child nodes of said task, i.e., they propagate much like nles propagate.
3. Commitments are motivated by non-local-effects or interactions between tasks.
4. Commitments have value to the agent and incur costs. But, the value provided by commitments is not quality as they do not contribute directly to the agent's problem solving activities.
5. Commitments generally pertain to the performance of tasks and actions, thus they must be considered when an agent is selecting which actions to perform, i.e., during the intention formation stage.
6. Commitments may have a temporal scope, thus they must also be considered when the agent is sequencing its intentions or scheduling.

Because of properties 1, 5, and 6, it is important to either have the effects of the commitment value reflected in the primitive actions OR to have the commitments somehow local to the actions so that they can be considered during intention formation and intention sequencing. Note: this does not necessarily mean to propagate the effects of the commitments (their value) to the quality, cost, and duration of actions in question. [2]

[2] Note that from a scheduling perspective, commitments and nles have something in common. Both must somehow be considered during the alternative generation phase and during sequencing. However, it is impossible to understand the full ramifications of either commitments or nles without actually building schedules (sequencing actions). The alternative generation phase could be integrated with the sequencing phase – that is we spend time sequencing alternatives as we propagate them from the leaves to the root. This would be a waste of computational resources in task structures that have no nles or commitments (and would be generally intractable for real problems), however, it might prove beneficial in situations where the optimistic estimates provided by ignoring the sequencing issues during alternative generation lead to a set of unscheduleable alternatives (i.e., more highly constrained situations).

One possible approach is to closely associate commitments with actions. Note: the approach I am about to describe separates the issue of when to re-evaluate the value assigned to commitments from the issue of when to re-evaluate whether or not to maintain a commitment. Figure 8.4 illustrates a transformation approach to dealing with commitments. When a commitment is given from Agent A to Agent B to provide the output of its grind limbs task to Agent B, the method is transformed from a "plain method" to a "committed method." The transformation results in two new methods replacing the original method – one method that acts as a placeholder for performing the action and *satisfying the commitment*, i.e., it models the benefit of maintaining and satisfying the commitment. The other method represents performing the action but *not satisfying* the commitment, i.e., it represents the cost associated with decommitment. (Note, since I've rewritten this, there is a glaring omission here. It is dealt with shortly.) This provides a mechanism to reason about commitments in the same fashion that we reason about other types of actions, however, it is somewhat misleading because just like the potential quality, potential cost, and potential duration characterizations of methods involved with nles, the satisfy-commitment-case and don't-satisfy-commitment-case actions are approximations. Even if the agent chooses to attempt to satisfy all committed actions, it may find at schedule time that it is impossible to sequence these actions due to other constraints, like nles between the actions. In this case, the unsatisfied actions must be replaced with the decommitment version of the action and sequenced in that way.

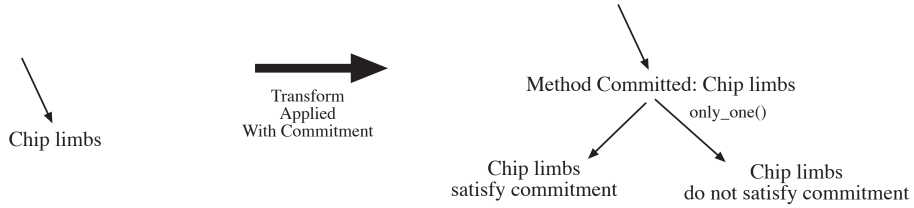


Figure 8.4: Representing Committed Actions Differently

Note, none of these thoughts addresses the issue of relating the value of a commitment with the q,d,c associated with tasks and actions. Perhaps the answer is as the answer above – adding a new attribute (or set of attributes) that denotes contribution to some measures of organizational utility. This mapping concept is also not perfect, in fact, it has one glaring hole. When you produce a schedule that includes neither the commitment maintaining version of the action or the commitment failing version of the action, then a hidden cost is incurred in whatever schedule is produced because the committed action was not performed, thus, there is a decommitment penalty that must be considered and that penalty is implicit in this case. (If the action was scheduled but not in such a way to satisfy the commitment, then that action would have carried the decommitment penalty along with it.) An alternative to the implicit decommitment business is to change the mapping of the task structure so that three methods are produced: 1) perform the action and satisfy the commitment, 2) perform the action but don't satisfy the commitment, 3) don't perform the action and thus decommit (this "method" would have zero duration, though it would incur the value penalty, whatever we decide "value" with respect to commitments means). To get around the implicit problem, however, you would also have to modify the task structure in such a way that one of the triple must be explicitly selected. This too is unappealing as the modification would have to pass all the way up the tree. Alternatively, we could split it out and each time a commitment is made, include a separate task under which live all decommitment cases..though this isn't right either, it suffers from the same problem as Figure 8.3 above.

There is a common thread here – we need some place other than the task structure to represent and model these things, but, at the same time, all of the issues discussed in this document relate to, or are closely associated with, TÆMS tasks and actions. The end result of this brainstorming may well be that none of these ideas are quite correct and that we do indeed need a different model or a separate hierarchy to use for commitments and the like.

For intellectual concepts only – please consult the dissertation for complete research

Bibliography

[1] Martin Andersson and Tumas Sandholm. Leveled commitment contracting among myopic individually rational agents. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, 1998.

[2] Martin Andersson and Tumas Sandholm. Leveled commitment contracts with myopic and strategic agents. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 38–44, 1998.

[3] Mihai Barbuceanu. Agents that work in harmony by knowing and fulfiling their obligations. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 89–96, 1998.

[4] Mihai Barbuceanu and Mark S. Fox. COOL: A language for describing coordination in multi agent syste ms. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS95)*, pages 17–25, 1995.

[5] Mihai Barbuceanu, Tom Gray, and Serge Mankovski. Coordinating with obligations. In *Proceedings of the Second International Conference on Autonomous Agents (Agents98)*, pages 62–69, 1998.

[6] Brett Benyo, Bryan Horling, Anita Raja, , Regis Vincent, Thomas Wagner, Ping Xuan, and Shelly XQ Zhang. The umass intelligent home project. http://mas.cs.umass.edu/research/mass/cs691v.html, 1998.

[7] M.E. Bratman. *Intention Plans and Practical Reason*. Harvard University Press, Cambridge, MA, 1987.

[8] Jonathan Bredin, David Kotz, and Daniela Rus. Market-based control for mobile agents. In *Proceedings of the Second International Conference on Autonomous Agents (Agents98)*, pages 197–204, 1998.

[9] Cristiano Castelfranchi. Commitments: From individual intentions to groups and organizations. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS95)*, pages 41–48, 1995.

[10] Phillip R. Cohen, Adam Cheyer, Michelle Wang, and Soon Cheol Baeg. An open agent architecture. In Michael N. Huhns and Munindar P. Singh, editors, *Readings in Agents*, pages 197–204. Morgan Kaufmann, 1998.

[11] P.R. Cohen and H.J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(3):213–261, 1990.

[12] P.R. Cohen and H.J. Levesque. Rational interactions as the basis for communication. In P.R. Cohen, J. Morgan, and M.E. Pollack, editors, *Intentions in Communication*. MIT Press, 1990.

[13] P.R. Cohen and H.J. Levesque. Communicative actions for artificial agents. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS95)*, page 445, 1995.

[14] P.R. Cohen and C.R. Perrault. Elements of a plan-based theory of speech acts. In Alan H. Bond and Les Gasser, editors, *Distributed Artificial Intelligence*, pages 169–186. Morgan Kauffmann, 1988.

[15] D. D. Corkill. *A framework for organizational self-design in distributed problem solving networks*. PhD thesis, University of Massachusetts at Amherst, Amherst, Massachusetts, February 1983.

[16] Daniel D. Corkill and Victor R. Lesser. The use of meta-level control for coodination in distributed problem solving networks. In *Proceedings of the Eigths International Joint Conference on Artificial Intelligence*, pages 748–756, 1983.

[17] Daniel D. Corkill and Victor R. Lesser. The use of meta-level control for coordination in a distributed problem solving network. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 748–755, Karlsruhe, Germany, August 1983.

[18] T. Dean and M. Boddy. An analysis of time-dependent planning. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 49–54, St. Paul, Minnesota, August 1988.

[19] K. Decker, A. Pannu, K. Sycara, and M. Williamson. Designing behaviors for information agents. In *Proceedings of the 1st Intl. Conf. on Autonomous Agents*, pages 404–413, Marina del Rey, February 1997.

[20] K. Decker, M. Williamson, and K. Sycara. Intelligent adaptive information agents. *Journal of Intelligent Information Systems*, 9:239–260, 1997.

[21] Keith Decker and Jinjiang Li. Coordinated hospital patient scheduling. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, pages 104–111, 1998.

[22] Keith S. Decker. *Environment Centered Analysis and Design of Coordination Mechanisms*. PhD thesis, University of Massachusetts, 1995.

[23] Keith S. Decker. TÆMS: A framework for analysis and design of coordination mechanisms. In G. O'Hare and N. Jennings, editors, *Foundations of Distributed Artificial Intelligence*, chapter 16. Wiley Inter-Science, 1995.

[24] Keith S. Decker and Victor R. Lesser. Generalizing the partial global planning algorithm. *International Journal of Intelligent and Cooperative Information Systems*, 1(2):319–346, June 1992.

[25] Keith S. Decker and Victor R. Lesser. Quantitative modeling of complex environments. *International Journal of Intelligent Systems in Accounting, Finance, and Management*, 2(4):215–234, December 1993.

[26] Keith S. Decker and Victor R. Lesser. Quantitative modeling of complex environments. *International Journal of Intelligent Systems in Accounting, Finance, and Management*, 2(4):215–234, December 1993. Special issue on "Mathematical and Computational Models of Organizations: Models and Characteristics of Agent Behavior".

[27] Keith S. Decker and Victor R. Lesser. Designing a family of coordination algorithms. In *Proceedings of the Thirteenth International Workshop on Distributed AI*, pages 65–84, Seattle, WA, July 1994. AAAI Press Technical Report WS-94-02. Also UMass CS-TR-94-14. To appear, Proceedings of the First International Conference on Multi-Agent Systems, San Francisco, AAAI Press, 1995.

[28] Keith S. Decker and Victor R. Lesser. Coordination assistance for mixed human and computational agent systems. In *Proceedings of Concurrent Engineering 95*, pages 337–348, McLean, VA, 1995. Concurrent Technologies Corp. Also available as UMASS CS TR-95-31.

[29] Keith S. Decker and Victor R. Lesser. Designing a family of coordination algorithms. In *Proceedings of the First International Conference on Multi-Agent Systems*, pages 73–80, San Francisco, June 1995. AAAI Press. Longer version available as UMass CS-TR 94–14.

[30] Robert Doorenbos, Oren Etzioni, and Daniel Weld. A scalable comparision-shopping agent for the world-wide-web. In *Proceedings of the First International Conference on Autonomous Agents*, pages 39–48, Marina del Rey, California, February 1997.

[31] Edmund H. Durfee. The distributed artificial intelligence melting pot. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1301–1306, 1991.

[32] Edmund H. Durfee and Thomas A. Montgomery. Coordination as distributed search in a hierarchical behavior space. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(6):1363–1378, 1991.

[33] E. Ephrati and J.S. Rosenschein. Divide and conquer in multi-agent planning. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 375–380, Seattle, 1994. AAAI Press/MIT Press.

[34] Eithan Ephrati and Jeffrey S. Rosenschein. The clarke tax as a consensus mechanism among automated agents. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 173–178, 1991.

[35] Oren Etzioni. Moving up the information food chain: Employing softbots on the world wide web. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1322–1326, Portland, OR, August 1996.

[36] Michael Georgeff. Communication and interaction in multiagent planning. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 125–129, August 1983.

[37] Michael Georgeff. A theory of action for multiagent planning. In *Proceedings of the Fourth National Conference on Artificial Intelligence*, pages 121–125, August 1984.

[38] B. Grosz and S. Kraus. Collaborative plans for group activities. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, Chambéry, France, August 1993.

[39] Barbara J. Grosz. Collaborative systems. *AI Magazine*, 17(2):67–85, 1996. AAAI-94 Presidential Address.

[40] Barbara J. Grosz and Sarit Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86:269–357, 1996.

[41] T.R. Gruber. Ontolingua: A mechanism to support portable ontologies. Technical Report KSL-91-66, Knowledge Systems Laboratory, Stanford University, 1992.

[42] T.R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. Technical Report KSL-93-4, Knowledge Systems Laboratory, Stanford University, 1993.

[43] Thomas Haynes, Sandip Sen, Neeraj Arora, and Rajani Nadella. An automated meeting scheduling system that utilizes user preferences. In *Proceedings of the First International Conference on Autonomous Agents (Agents97)*, pages 308–316, 1997.

[44] Randall Hill, Johnny Chen, Jonathan Gratch, Paul Rosenbloom, and Milind Tambe. Intelligent Agents for the Synthetic Battlefield: A Company of Rotary Wing Aircraft. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 1006–1012, July 1997.

[45] Eric Horvitz, Gregory Cooper, and David Heckerman. Reflection and action under scarce resources: Theoretical principles and empirical study. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, August 1989.

[46] Eric Horvitz and Jed Lengyel. Flexible Rendering of 3D Graphics Under Varying Resources: Issues and Directions. In *Proceedings of the AAAI Symposium on Flexible Computation in Intelligent Systems*, Cambridge, Massachusetts, November 1996.

[47] Marcus J. Huber and Edmund H. Durfee. Deciding when to commit to action during observation-based coordination. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS95)*, pages 163–170, 1995.

[48] Michael N. Huhns and Munindar P. Singh. Agents and multiagent systems: Themes, approaches, and challenges. In Michael N. Huhns and Munindar P. Singh, editors, *Readings in Agents*, pages 1–23. Morgan Kaufmann, 1998.

[49] Nicholas R. Jennings, Katia Sycara, and Michael Wooldridge. A roadmap of agent research and development. *Autonomous Agents and Multi-Agent Systems*, 1(1):8–38, 1998.

[50] N.R. Jennings and E.H. Mamdani. Using joint responsibility to coordinate collaborative problem solving in dynamic environments. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 269–275, 1992.

[51] N.R. Jennings and M. Wooldridge. Applications of intelligent agents. In Nicholas R. Jennings and Michael J. Wooldridge, editors, *Agent Technology*, pages 3–28. Springer, 1998.

[52] Stanley M. Sutton Jr. and Leon J. Osterweil. The Design of a Next-Generation Process Language. UMASS Department of Computer Science Technical Report TR-96-30, 1996. Revised January 22, 1997.

[53] Gal A. Kaminka and Milind Tambe. What is Wrong With Us? Improving Robustness Through Social Diagnosis. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, July 1998.

[54] Henry Kautz, Bart Selman, Michael Coeh, Steven Ketchpel, and Chris Ramming. An experiment in the design of software agents. In Michael N. Huhns and Munindar P. Singh, editors, *Readings in Agents*, pages 125–130. Morgan Kaufmann, 1998.

[55] Frank Klassner. *Data Reprocessing in Signal Understanding Systems*. PhD thesis, University of Massachusetts at Amherst, Amherst, Massachusetts, September 1996.

[56] Kazuhiro Kuwabara, Toru Ishida, and Nobuyasu Osato. Agentalk: Coordination protocol description for multiagent systems. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS95)*, page 445, 1995.

[57] Kazuhiro Kuwabara, Toru Ishida, and Nobuyasu Osato. Agentalk: Describing multiagent coordination protocols with inheritance. In *Proceedings of the 7th International Conference on Tools with Artificial Intelligence (ICTAI-95)*, pages 460–465, 1995.

[58] Yannis Labrou and Tim Finin. A Proposal for a new KQML Specification. Computer Science Technical Report TRCS-97-03, University of Maryland Baltimore County, February 1997.

[59] Victor Lesser, Keith Decker, Norman Carver, Alan Garvey, Daniel Neiman, Nagendra Prasad, and Thomas Wagner. Evolution of the GPGP Domain-Independent Coordination Framework. Computer Science Technical Report TR-98-05, University of Massachusetts at Amherst, January 1998.

[60] Victor Lesser, Bryan Horling, Frank Klassner, Anita Raja, Thomas Wagner, and Shelley XQ. Zhang. BIG: A resource-bounded information gathering agent. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, July 1998. See also UMass CS Technical Reports 98-03 and 97-34.

[61] Victor R. Lesser. Reflections on the nature of multi-agent coordination and its implications for an agent architecture. *Autonomous Agents and Multi-Agent Systems*, 1(1):89–111, 1998.

[62] Thomas W. Malone and Kevin Crowston. Toward an interdisciplinary theory of coordination. Center for Coordination Science Technical Report 120, MIT Sloan School of Management, 1991.

[63] S. H. Nawab and E. Dorken. Quality versus Efficiency Tradeoffs in STFT Computation. *IEEE Transactions on Signal Processing*, 43(2):998–1002, 1995.

[64] S. H. Nawab and T. Quatieri. Short-time Fourier transform. In S. H. Nawab and T. Quatieri, editors, *Advanced Topics in Signal Procesing*. Prentice-Hall, New Jersey, 1988.

[65] Timothy J. Norman, Carles Sierra, and Nick R. Jennings. Rights and commitment in multi-agent agreements. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, 1998.

[66] M.V. Nagendra Prasad and V.R. Lesser. Learning situation-specific coordination in generalized partial global planning. In *AAAI Spring Symposium on Adaptation, Co-evolution and Learning in Multiagent Systems*, Stanford, March 1996.

[67] Daniela Rus, Robert Gray, and David Kotz. Transportable information agents. In *Proceedings of the First International Conference on Autonomous Agents (Agents97)*, pages 228–236, 1997.

[68] Stuart J. Russell and Shlomo Zilberstein. Composing real-time systems. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 212–217, Sydney, Australia, August 1991.

[69] Tumas Sandholm and Victor Lesser. Issues in automated negotiation and electronic commerece: Extending the contract net framework. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS95)*, 1995.

[70] Yoav Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–93, March 1993.

[71] Herbert A. Simon. *Administrative Behavior*. Macmillan Company, New York, NY, 1945.

[72] Herbert A. Simon. *Models of Bounded Rationality*. MIT Press, Cambridge, MA, 1982.

[73] Kilian Stoffel, Merwyn Taylor, and Jim Hendler. Efficient management of very large ontologies. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 442–447, July 1997.

[74] Toshi Sugawara and Victor R. Lesser. Learning to improve coordinated actions in cooperative distributed problem-solving environments. *Machine Learning*, 1998.

[75] Toshiharu Sugawara and Victor R. Lesser. On-line learning of coordination plans. Computer Science Technical Report 93–27, University of Massachusetts, 1993.

[76] Milind Tambe. Teamwork in Real-world, Dynamic Environments. In *Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS96)*, pages 361–368, 1996.

[77] Milind Tambe. Tracking Dynamic Team Activity. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 80–87, July 1996.

[78] Milind Tambe. Agent Architectures for Flexible, Practical Teamwork. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 22–28, July 1997.

[79] Milind Tambe and Weixiong Zhang. Towards Flexible Teamwork in Persistent Teams. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, 1998.

[80] Millind Tambe. Implementing Agent Teams in Dynamic Multi-agent Environments. *Journal of Applied Artificial Intelligence*, 1997.

[81] Millind Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[82] Manuela Veloso, Peter Stone, and Kwun Han. The CMUnited-97 robotic soccer team: Perception and multiagent control. In *Proceedings of the Second International Conference on Autonomous Agents (Agents98)*, pages 78–85, 1998.

[83] Thomas Wagner, Alan Garvey, and Victor Lesser. Complex Goal Criteria and Its Application in Design-to-Criteria Scheduling. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 294–301, July 1997. Also available as UMASS CS TR-1997-10.

[84] Thomas Wagner, Alan Garvey, and Victor Lesser. Design-to-Criteria Scheduling: Managing Complexity through Goal-Directed Satisficing. In *In the Proceedings of the AAAI-97 Workshop on Building Resource-Bounded Reasoning Systems*, July 1997. Also available as UMASS CS TR-97-16.

[85] Thomas Wagner, Alan Garvey, and Victor Lesser. Leveraging Uncertainty in Design-to-Criteria Scheduling. UMASS Department of Computer Science Technical Report TR-97-11, January, 1997.

[86] Thomas Wagner, Alan Garvey, and Victor Lesser. Criteria-Directed Heuristic Task Scheduling. *International Journal of Approximate Reasoning, Special Issue on Scheduling*, 19(1-2):91–118, 1998. A version also available as UMASS CS TR-97-59.

[87] Thomas Wagner and Victor Lesser. Design-to-Criteria Scheduling for Intermittent Processing. UMASS Department of Computer Science Technical Report TR-96-81, November, 1996.

[88] Thomas Wagner, Victor Lesser, Brett Benyo, Anita Raja, Ping Xuan, and Shelly XQ Zhang. GPGP2: Improvement Through Divide and Conquer. Working document, 1998.

[89] M.P. Wellmen, E.H. Durfee, and W.P. Birmingham. The digital library as community of information agents. *IEEE Expert*, June 1996.

[90] J. E. White. Telescript technology: The foundation for the electronic marketplace. General Magic White Paper, General Magic Inc., 1994.

[91] Michael Wooldridge and Nicholas R. Jennings. Pitfalls of agent-oriented development. In *Proceedings of the Second International Conference on Autonomous Agents (Agents98)*, pages 385–391, 1998.

[92] M.J. Wooldridge and N.R. Jennings. Agent theories, architectures, and languages: A survey. In *Proceedings of the Workshop on Agent Theories, Architectures, and Languages (ECAI)*, pages 1–32, 1994.

[93] M.J. Wooldridge and N.R. Jennings. Formalizing the cooperative problem solving process. In *Thirteenth International Workshop on Distributed Artificial Intelligence*, pages 403–417, July 1994.

[94] Ping Xuan, Brett Benyo, Bryan Horling, Anita Raja, , Regis Vincent, Thomas Wagner, and Shelly XQ Zhang. Coordination protocols for intelligent home-control agents. http://mas.cs.umass.edu/research/mass/homeCoord.html, 1998.

[95] S. Zilberstein and S. J. Russell. Optimal composition of real-time systems. *Artificial Intelligence*, 82(1):181–214, December 1996.

[96] Shlomo Zilberstein. Using anytime algorithms in intelligent systems. *AI Magazine*, 17(3):73–83, 1996.

[97] M. Zweben, B. Daun, E. Davis, and M. Deale. Scheduling and rescheduling with iterative repair. In M. Zweben and M. Fox, editors, *Intelligent Scheduling*, chapter 8. Morgan Kaufmann, 1994.