Directing Agent Communities: An Initial Framework

Karen L. Myers David N. Morley
Artificial Intelligence Center
SRI International
333 Ravenswood Ave.,
Menlo Park, CA 94025
myers@ai.sri.com morley@ai.sri.com

Abstract

For effective use of agent communities in extended problem-solving sessions, humans must be able to both guide agent operations and understand agent progress. This paper presents a framework for *directability* of a community of agents by a human supervisor that focuses on three dimensions: adjustable agent autonomy, strategy preference, and community-level constraints. The paper also describes a prototype multiagent system that applies these concepts of directability within an intelligence-gathering domain.

Introduction

The technical and public press are filled these days with visions of a not-too-distant future in which humans rely on software and hardware agents to assist with problem solving in environments both physical (e.g., smart offices, smart homes) and virtual (e.g., the Internet). The notion of *delegation* plays a central role in these visions, with humans offloading responsibilities to agents that can perform activities in their place.

Successful delegation requires more than the assignment of tasks. A good manager will generally provide directions to a subordinate so that tasks are performed to his or her liking. To ensure effectiveness, he or she will monitor the progress of subordinates, occasionally interrupting to provide advice or to resolve problems.

The agents research community has, for the most part, focused on the mechanics of building autonomous agents and techniques for communication and coordination among agents. In contrast, little attention has been paid to supporting human interactions with agents of the type required for extended problem-solving sessions. Most agent frameworks fall at the extremes of the interaction spectrum, either assuming full automation by the agents with no means for user involvement, or requiring human intervention at each step along the way. Recently, however, there has been increased interest in agent systems designed specifically to support human interaction (Ferguson & Allen 1998; Pynadath *et al.* 1999; Cesta, D'Aloisi, & Collia 1999; Bonasso 1999).

This paper describes an ongoing research project focused on supporting user directability of a team of agents for solving complex problems. Specifically, we are interested in extended problem-solving sessions that involve numerous tasks, each requiring specialized activities on the part of agents for their accomplishment. Tasks may be only partially defined and could evolve over time in response to partial results or changes in operating conditions. The environment itself will be dynamic and unpredictable, thus precluding prespecified solutions to assigned tasks.

Frameworks to support human-agent problem solving within this type of setting will require the following characteristics:

Flexible Task Delegation Humans must be able to assign tasks by using powerful, high-level languages. Task specifications would provide general descriptions of actions to be performed rather than detailed algorithms.

Adaptability Agents and agent control mechanisms must be adaptable to unexpected events and changes in task requirements.

Mixed-Initiative Control Humans and agents must be able to work cooperatively, in a manner that complements their respective strengths.

Visibility into Agent Activity Humans must have visibility into the progress of agents on delegated tasks.

We are developing a framework called Taskable Reactive Agent Communities (TRAC) that supports directability of a team of agents by a user. Within TRAC, the user assigns tasks to agents along with guidance that imposes boundaries on agent behavior. During execution, the human manages agent activity in accord with

a level of involvement that suits his or her needs. In essence, our work can be viewed as providing a form of process management technology that enables human control of agent communities.

Our aims for the TRAC project are to identify appropriate types of guidance for directing agents, to design languages for expressing such guidance, and to develop techniques for operationalizing guidance into influences on agent behavior. Issues related to human-computer interaction, while important, are beyond our current scope.

The paper also describes a prototype system, called TIGER, which instantiates the TRAC approach to directability for the application of multiagent intelligence gathering in response to a natural disaster. TIGER provides control over a collection of simulated physical assets each embodied as a separate agent. These physical assets can be tasked to perform a range of actions related to intelligence gathering, and to provide assistance with eventualities such as medical emergencies, evacuations, and infrastructure repair. Although in its early stages, TIGER provides a directability environment in which humans can delegate tasks to agents while providing guidance to control their behavior.

Agent Operations Model

We adopt a typical Belief-Desire-Intention (BDI) framework in the style of (Rao & Georgeff 1995), whereby an agent undertakes actions to address its desires, relative to its current beliefs about the operating environment.

A set of *plans* defines the range of activities that an agent can perform to respond to events or to achieve assigned tasks; our plan model is based on (Wilkins & Myers 1995). Plans are parameterized templates of activities that may require variable instantiations to apply to a particular situation. The *cue* of a plan defines the reason for invoking a plan. A plan may be activated to respond to either a new goal (originating with the agent itself, another agent, or a human supervisor) or to a perceived event corresponding to a change in the agent's beliefs about the world state. *Preconditions* associated with plans define gating constraints that must be satisfied in order for a plan to be applied. A plan is said to be *applicable* to a world change (e.g., new goal or event) when the plan cue matches the stimulus, and the plan preconditions are satisfied in the current world state. Plans can be decomposed into other plans (thus providing a hierarchical model of activity), or can specify actions that agents can execute directly.

An agent's plan library will generally contain a range of plans describing alternative responses to posted goals or events. Sets of these plans may be *operationally equivalent* (i.e., they share the same cue and preconditions) but differ in the approach that they embody. In systems such as PRS (Georgeff & Ingrand 1989), control policies for selecting among operationally equivalent alternatives can themselves be encoded as plans.

A BDI interpreter runs a continuous *sense-decide-act* loop to respond to changes in its operating environment. At the start of each cycle, the interpreter collects all new goals and events (i.e., changes in its beliefs about the world). Next, it determines whether there are any plans in its library that are applicable to these changes. From this set, it selects some subset for execution and creates intentions for them. Finally, some bounded number of steps for each current intention are performed. Within this paradigm, agents make three main classes of decisions:

D1 whether to respond to new goals and events

D2 how to select among multiple applicable plans

D3 how to select instantiations for plan variables

Our directability framework assumes that agents are capable of fully autonomous operation. More concretely, an agent's plan library covers the range of activities required to perform its assigned tasks. This assumption means that agents do not depend on the human supervisor to provide knowledge for task execution. Within this setting, guidance provides customization of agent behavior to suit the preferences of the human supervisor. In many applications, such guidance will enable superior performance, given that few plan libraries will match the experience, breadth of knowledge, and reasoning capabilities that a human supervisor can bring to the decision-making process.

Agent Directability

Our model of agent directability emphasizes general and task-specific recommendations to influence the activities undertaken by agents in their execution of assigned tasks. We focus on the complementary capabilities of *flexible delegation of tasks* and *visibility into operations*.

For flexible delegation, we are developing techniques by which a human can both assign tasks to an agent community and impose restrictions on the activities that agents undertake while performing those tasks. Our focus has been in three main areas: (a) adjustable levels of agent autonomy, (b) strategy preferences that describe the approaches to be used by an individual agent in executing assigned tasks, and (c) community-level behavior constraints.

Successful delegation further requires human visibility into agent activities. For this reason, our framework includes methods for *customizable reporting*, which allow agents to adjust the frequency and detail of information that they transmit regarding their status and progress on assigned tasks.

This section presents our overall approach to agent directability; the following section describes an initial instantiation of the approach for a multiagent intelligence-gathering application.

Adjustable Autonomy

We define the autonomy of an agent to be the extent to which it is allowed to make decisions (specifically, D1 – D3) on its own initiative. In situations where activities are routine and decisions straightforward, a human may be content to delegate all problem-solving responsibility to an agent. However, in situations where missteps could have severe consequences, a human may wish to assert greater control over agent decision making. For this reason, the degree of autonomy of an individual agent should necessarily be controllable by a human.

Because we are interested in domains where agents will need to operate with high degrees of autonomy, we assume a *permissive* environment: unless stated otherwise, agents are allowed to operate independent of human interaction. Our approach allows the human to adjust the scope of operations that can be undertaken by an agent on its own terms, focusing on the notions of *permission requirements* for action execution and *consultation requirements* for decision making.

Permission Requirements Permission requirements declare conditions under which an agent must elicit approval from the human supervisor before executing actions. For example, the permission requirement

> Obtain permission before abandoning survey tasks with Priority $>$ 3. (G1)

imposes the constraint that an agent request approval from its supervisor to abandon a certain class of tasks.

Consultation Requirements Consultation requirements designate a class of agent decisions that should be deferred to the human supervisor. These decisions may relate either to action selection (G2) or variable instantiation (G3):

> Consult when determining a response to survey task failure. (G2)
>
> Consult when selecting locations for staging bases. (G3)

Our model of permission and consultation requirements, like earlier work on authority models, provides a mechanism to block performance of certain actions by an agent. However, authority models are generally static (e.g., the *levels of autonomy* in (Bonasso 1999)) and often derived from organizational structures. In contrast, our approach aims to provide a rich language for expressing permission and consultation requirements, which can vary throughout a problem-solving session.

Strategy Preference

Guidance for *strategy preferences* expresses recommendations on how an agent should accomplish a particular task. These preferences could indicate specific plans to employ or restrictions on plans that should not be employed, as well as constraints on how plan variables can be instantiated. This form of guidance can be used by a human to tailor the behavior of an agent at runtime.

For example, a human directing a collection of intelligence-gathering agents might declare the following sorts of strategy preferences:

> Try contacting Nongovernmental Organizations for information before sending vehicles to the towns on the west coast. (G4)
>
> Only use helicopters for survey tasks in sectors that are expected to be inaccessible by truck for more than 1 week. (G5)
>
> Maintain 2 helicopters in reserve for emergency evacuations. (G6)

The first directive expresses a preference for selecting among operationally equivalent plans; the second restricts the choice of resource types for instantiating certain plan variables; the third imposes a constraint that indirectly impacts plan selection options.

Collective Activity

The forms of directability described above focus on influencing the behavior of an individual agent. Users will also want to express control at the *community* level, to encourage or discourage various types of collective or emergent behaviors. For example, operationalization of the directive

> Keep 2 trucks within 15 miles of headquarters. (G7)

would require ongoing cooperation among the available truck agents to avoid violation.

Customizable Reporting

The degree of visibility required by a human into the activity of a particular agent will necessarily vary according to circumstances. During initial phases, periodic detailed reports may be desired to provide a clear overall picture of operations. As crises arise, immediate reports on the status of certain activities may become essential; in addition, detailed status reports could be replaced by high-level summaries since the human already has a good understanding of the lay of the land.

Our model of *customizable reporting* enables a user to tailor the amount and type of information produced

by agents to his or her evolving needs. We define different classes of reporting methods that can be activated by the human as appropriate. Dimensions of adjustability include the context in which events should be reported, frequencies for periodic reports, and level of detail.

The TIGER System

We have developed a prototype system, called TRAC Intelligence Gathering and Emergency Response (TIGER), to serve as a testbed for exploring our ideas on agent directability. TIGER focuses on agent directability for an intelligence-gathering domain. The current version of TIGER, built on top of PRS (Georgeff & Ingrand 1989), explores instantiations of our models for adjustable autonomy, strategy preference, and customizable reporting.

TIGER serves as part of a *disaster response team* whose objective is to provide humanitarian relief in the wake of a natural disaster. Other organizations within the team provide logistics (e.g., supplies distribution), operations (e.g., repair of infrastructure), and medical services. These different organizations have a number of physical assets (trucks and aircraft) available for their use. As would be expected, these organizations must share information and resources to perform their functions effectively. A human commander oversees operations, dynamically tasking organizations to implement the relief process.[1]

The primary role for TIGER is to gather information in response to requests initiated by other members of the disaster response team. These requests can result in tasks to seek out information such as the current state of infrastructure (roads, bridges) in designated regions, or to collect supply requirements (medical, food, water, shelter) of designated population centers within impacted regions. There can also be requests to be informed of key events (such as medical emergencies) as they become known.

Beyond information gathering and dissemination, the TIGER agent community must also respond to certain types of unexpected events (e.g., participating in evacuations, assisting with medical emergencies). These responsibilities make it necessary for TIGER agents to incorporate reactive capabilities that balance responsiveness with ongoing goal attainment.

The scope and complexity of the intelligence-gathering operations within the disaster relief context preclude management of agent operations by a human commander. However, effective coordination of the available assets requires human supervision. As such, this domain provides an excellent example of an application that will benefit from technology for agent directability.

[1]The system operates within a testbed that simulates a major hurricane in Central America; the simulation is built on the MAPLE system (`www.cs.cmu.edu/~maple/`).

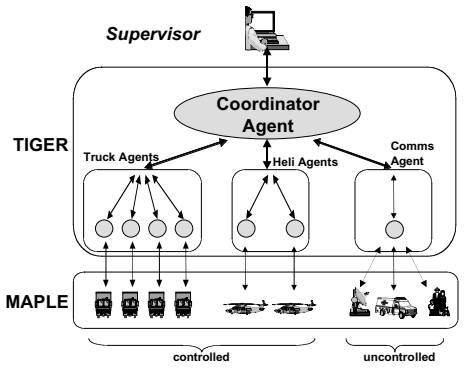

Figure 1: TIGER Architecture

Agent Community Organization

Figure 1 displays the organization of agents within TIGER. The system has a collection of simulated *physical agents* (trucks and helicopters) at its disposal that can be used to gather information and respond to emergencies. In addition, there is a set of simulated *communications agents* (other relief organizations, nongovernment organizations, local officials) that can be consulted to obtain information. TIGER contains a separate controller for each of the physical agents, as well as a communications manager for interacting with the various simulated communications agents. We refer to these controller agents as the *task execution agents* within TIGER, because they instigate and manage the activities required to perform assigned tasks.

The *coordinator agent* provides global management of tasks within the community, acting as a mediator between the human supervisor and the task execution agents. It also manages interactions with members of the disaster response team who request information (i.e., its *information clients*).

The set of task execution agents in the TIGER community can be changed by external forces. For example, the logistics team may request to borrow some number of trucks to support delivery of critical supplies for a short period of time. Alternatively, when inundated with high-priority tasks that require immediate response, the intelligence team may request additional physical assets to join its agent community temporarily.

Within TIGER, agents behave cooperatively when they believe it is in the best interest of the overall com-

munity. Hence, agents generally respond to the direction of the human supervisor. Agents can behave in ways that seem uncooperative (e.g., refusing a request to adopt a task because its current task has higher priority), but this is due to agents having different information rather than conflicting interests.

Tasking Model

The coordinator agent places incoming task requests into a pool of tasks waiting to be performed. It also maintains a pool of currently unallocated agents that are available to perform tasks. While there are tasks to be performed and agents unallocated, the coordinator agent matches a task with an execution agent based on properties of the tasks, the agents, and the current knowledge about the state of the roads and bridges. Task properties include *location*, *priority* (an integer from 0 to 10), *type* (e.g., survey, rescue), *deadline* (for completing the task), and *status* (e.g., pending, completed, failed). The agent properties include *agent type* (helicopter or truck) and *location*. After making the match, the coordinator agent requests that the execution agent perform the task.

For simplicity, we limit each task execution agent to at most one active task at any point in time. Agents may also have pending tasks (which they intend to undertake) and preempted tasks (which were begun but put aside for higher-priority tasks). Tasks are assigned to individual agents and do not require any coordination with other agents for their completion.

Unexpected events, such as a medical emergency, may require immediate response. Events are characterized by the properties *location*, *time* (of the event), *severity* (an integer 0 to 10), *number of people affected*, and *type* (e.g., evacuation, medical). Rather than creating a new task for the task pool, the coordinator agent selects an appropriate task execution agent to deal directly with each event.

These characteristics of tasking simplify the decision process for what an execution agent should do when it receives a task request. The agent can choose among several combinations of the following actions, each of which is implemented as a different plan:

ignore ignore the event

register create a new task to respond to the event

adopt adopt a new task to respond to the event

abandon abandon the current active task completely

postpone postpone the current active task while another is executed

transfer give up responsibility for the current active task by transferring it back to the coordinator agent

TIGER Guidance

An agent's plans specify the space of possible responses to new facts and events. However, there may be considerations beyond the reasoning capabilities of the agent that should determine a course of action. As such, the human supervisor of the agent community plays a vital role: providing guidance to direct agent activities. TIGER currently supports two types of guidance: *adjustable autonomy* and *strategy selection*.

Adjustable Autonomy Our model of adjustable autonomy focuses on deferring decisions for action selection and variable instantiation to a human supervisor. To this end, we have formulated a language for specifying classes of actions and decisions to be deferred.

Our approach builds on an *action metatheory*, which provides an abstracted characterization of actions that highlights key semantic differences. As discussed in (Myers 2000), a metatheory can provide a rich vocabulary for describing characteristics of actions, thus providing a powerful basis for supporting communication with users. The main concepts within our metatheory for adjustable autonomy are *features* and *roles* (similar in spirit to those of (Myers 1996)).

A *feature* designates an attribute of interest for a plan that distinguishes it from other plans that could be applied to the same task. For example, among plans that can be used for route planning, there may be one that is Optimal but Slow with a second that is Heuristic but Fast. Each of these attributes could be modeled as a feature. Thus, although the two plans may be operationally equivalent (i.e., same cue and preconditions), their intrinsic characteristics differ significantly. Features can be used to capture such semantics differences, thus providing the means to distinguish between operationally equivalent alternatives.

A *role* describes a capacity in which a domain object is to be used within a plan; it maps to an individual variable within a plan. For instance, a route determination plan may contain variables `location.1` and `location.2`, with the former corresponding to the Start and the latter a Destination.

Features can be used to define abstract classes of activity for an agent through the specification of a set of required and prohibited features. Thus, one can refer to the class of plans that have the feature Survey but not Heuristic. Roles can be used to refine such characterizations by requiring that a plan include roles such as Start and Destination, and imposing constraints on the variables to which the roles refer (e.g., requiring that the distance between Start and Destination not exceed some threshold). We refer to abstractions of this type as *activity specifications*.

Permission requirements are defined in terms of a *permission-constrained activity* and an *agent context*.

The former, expressed as an activity specification, designates a class of actions for which permission must be obtained. The agent context defines conditions on the operating state of the agent that limit the scope of the permission requirement; it is defined in terms of the three basic components of a BDI framework:

Beliefs constraints that represent conditions on the current world state

Desire a goal or event indicating the high-level motivation for which the action is being performed

Intention an activity specification for actions that the agent is currently executing

The interpretation of a permission requirement is that whenever an agent's current state matches the specified agent context, then any action under consideration for execution that matches the permission-constrained activity requires permission from the human supervisor.

Example 1 (Permission Requirement) The statement Seek permission to abandon survey tasks requested by the Ops Manager could be translated into a permission requirement of the form

```
Agent Context:
 Desire:
  (ACHIEVE (SURVEY-REQUEST location.1
                priority.1 OpsManager))

Permission-constrained Activity:
 Features: Abandon
 Roles: Current-Task
 Role Constraints:
    (= (TASK-TYPE Current-Task) SURVEY)
```

Consultation requirements are defined by an *agent context* and a *decision specification*. The agent context is defined as for permission requirements, while the decision specification is defined in terms of a metatheory *role*. The interpretation of a consultation requirement is that when an agent is in a situation that matches the agent context, any instantiation decision for a variable that matches the designated role should be passed to the human supervisor.

Example 2 (Consultation Requirement) The statement When in radio contact, consult when selecting locations for staging bases could be translated into a permission requirement of the form:

```
Agent Context:
 Belief: (IN-RADIO-CONTACT)

Decision Specification:
 Role: Staging-Base
```

The current TIGER system supports permission requirements for task execution agents, based on a simple action metatheory related to task management. For task management activity specifications, the candidate plans are tagged with some subset of the features {Ignore, Register, Adopt, Abandon, Postpone, Transfer}, depending upon the classes of task management activities that the plan involves. The key roles are Current-Task indicating the current task to be adopted, abandoned, or transferred, and New-Task indicating the potential task to be adopted, registered, or ignored.

The enforcement of permission requirements relies on specialized *metalevel plans* that control the selection and execution of other plans. The BDI interpreter cycle proceeds as normal to select a response to an event or goal (i.e., identifying applicable plans and ordering them according to any current control policies); at that point, however, permission requirements for the selected plan are checked and the supervisor is consulted (if necessary). If the supervisor denies permission, the interpreter then considers the next most highly ranked applicable plan. This process continues until either a permissible plan is identified, or the set of applicable plans has been exhausted.

Strategy Selection: Task Management Guidance Task management constitutes a major component of an execution agent's decision-making process. An execution agent must determine what to do if, while executing one task, the coordinator agent passes it a second task. The agent may abandon, postpone, or transfer the responsibility for its current task and start executing the new task. It may decide to continue its current task and either postpone the new task, send it back to the coordinator agent to place in the task pool, or just ignore it if it is not important enough. Similarly, agents must decide when to drop tasks that are not progressing well in favor of new tasks that have higher potential for success.

For TIGER, we have developed a language in which to express restrictions and preferences on the task management behaviors of execution agents. This language encompasses concepts such as the types of events to which agents should respond, priorities for potentially conflicting tasks, and control strategies for interleaving or sequencing partially completed and new activities. For example, the following guidance expresses a preferred response to certain types of medical emergencies.

> Respond to medical emergencies involving more than 10 people when the severity exceeds the current task priority. (G8)

As with adjustable autonomy, our approach to the specification of reactivity guidance relies on the action metatheory. The guidance is defined in terms of two components: the *agent context* and the *response*.

The agent context denotes conditions on the state of the agent for the guidance to apply. It includes *constraints* representing beliefs about the current world state, a specific *goal* or *event* triggering the action, and

an *activity specification* describing a class of actions that the agent must currently be executing.

The response consists of a *modality* and an *activity specification*. The modality indicates a preference to perform (or not) a class of activities (indicated by either *should* or *should-not*). The activity specification designates the category of response and, by imposing constraints on roles, may specify parameters of any new task undertaken in response to the triggering event.

Events are characterized by an *event class* (e.g., emergency, movement-failure) and a collection of *event properties*. Tasks are similarly classified by a *task class* (survey, rescue) and a collection of *task properties*. The context constraint is a logical expression over the event class and properties and the task class and properties.

Example 3 provides an illustration. The agent context for the guidance is defined by the `Event`, `Roles`, and `Constraints` slots. An emergency event of type `MEDICAL` triggers consideration of the guidance, provided the number of people affected is greater than 10 and the agent's current task (if any) has priority less than the severity of the event.

The activity specification of the response (indicated by the `Features`, `Roles`, and `Constraints` slots) is the class of actions where a new task, `New-Task`, to respond to the event is adopted and the priority of `New-Task` equals the severity of the event. The response modality `SHOULD` indicates that this class of actions should be preferred to alternatives.

Example 3 The statement (G8) could be represented by the following task mangement guidance.

```
Agent Context:
  Event: (EMERGENCY
           :location ELOCATION.1
           :type MEDICAL
           :severity ESEVERITY.1
           :nbr-affected ENUMBER.1
           :description EDESCRIPTION.1
           :simtime ETIME.1)
  Roles: Current-Task
  Constraints:
    (AND (> ENUMBER.1 10)
         (> ESEVERITY.1
            (TASK-PRIORITY Current-Task)))
Response:
  Modality: SHOULD
  Features: ADOPT
  Roles: New-Task
  Constraints:
   (= (TASK-PRIORITY New-Task) ESEVERITY.1)
```

Task management guidance provides user recommendations for an agent's response to events that match the specified agent context. However, other constraints may prevent execution of the recommended response. For example, the guidance in Example 3 recommends a response to a medical emergency but a lack of resources may prevent the adoption of any new tasks. On a related note, users may provide incompatible pieces of guidance to an agent that recommend mutually inconsistent responses.

To address these issues, we interpret guidance as an expression of preference over the set of applicable plans, and rely on an ordering over guidance rules to address conflicts. Currently, we consider only *direct* conflicts, which are defined in terms of opposing recommendations (e.g., *Do P* and *Don't do P*). More powerful detection mechanisms are required to deal with *indirect* conflicts, where the impact of one piece of guidance is incompatible with a second but only due to downstream effects. For example, the simultaneous execution of two plans could lead to deadlock or livelock situations.

As in the case of adjustable autonomy, we implement the task management guidance using specialized meta-level procedures. The set of alternative responses to an event is checked against the current guidance. If the agent context for a guidance rule matches the triggering event, the current world state, and the task currently being executed, then the response of the guidance rule is matched against each alternative response plan. Matching the response of a guidance rule to a plan consists of checking the features of the response action specification against the features associated with the plan and unifying the action specification roles with appropriate parameters of the plan. Direct conflicts are then resolved in accord with the guidance-ordering scheme that has been defined.

Customizable Reporting TIGER agents report on three types of information:

- *Task results:* information collected as part of an information-gathering task (e.g., state and needs of the people in various population centers)
- *State Information:* world-state properties that may be of general interest (e.g., emergencies, damage)
- *Task execution status* (e.g., inability to perform task)

The level of detail and rate of flow of this information need to be carefully tailored to balance informativeness against information overload.

Within TIGER, the human supervisor, as well as each client, has the ability to adjust the level of detail and timing of reports that it receives as individual needs and capabilities to deal with the information change. The client can request that reports about the status and/or results of tasks can be generated periodically or triggered by designated events (such as task execution problems, or a medical emergency).

Discussion

This paper describes a framework for human directability of agent communities, and an instantiation of those ideas in the TIGER system for multiagent intelligence gathering. Although in its early stages, TIGER already demonstrates the value of directability technology for helping users manage complex agent communities.

Many outstanding issues remain to be addressed. Our future work will focus on the following topics.

Detecting and Resolving Guidance Conflicts As discussed above, TIGER recognizes only direct conflicts among guidance that are rooted in opposing recommendations (i.e., for and against the same plan). Less explicit forms of conflict will require more powerful detection methods that reason about the downstream effects and requirements of plans. We are also interested in expanding our simple prioritization approach to conflict guidance to incorporate more advanced models for conflict resolution policies (e.g., (Dignum *et al.* 2000; Lupu & Sloman 1999)).

Community Guidance Our work to date has focused on guidance that influences the behavior of individual agents. A more general category of guidance may refer to behavior of the community as a whole (e.g., maintaining helicopters in reserve for emergencies). This type of guidance will require more sophisticated mechanisms that support coordinated action selection across groups of agents.

Behavior Extensions Our strategy preference guidance ranks previously defined alternatives; it does not expand the behavioral capabilities of the agent. An alternative approach would be to support guidance to specify *new* behaviors. Much of the work on *policy-based control* for distributed systems managements falls within this category (for e.g., (Moffett & Sloman 1993)). The ability to define new agent behaviors at runtime has much appeal as it enables the dynamic extension of agent capabilities. However, runtime behavior definition must be carefully controlled to prevent the creation of incorrect or dangerous agent capabilities.

Collaborative Control Our model of agent directability provides a form of *supervised autonomy* (Barber & Martin 1999) in which control over autonomy rests solely with the human supervisor. Some situations may benefit from a more collaborative approach (Fong, Thorpe, & Baur 1999), where both sides share control over initiative. For example, an agent may choose to initiate a dialogue with the human in situations where adherence to guidance would interfere with the pursuit of current goals, rather than blindly following the user's recommendations.

Acknowledgments The authors would like to thank Eric Hsu for his contributions in developing the TIGER interface. This research has been supported by DARPA Contract F30602-98-C-0160 under the supervision of Air Force Research Laboratory – Rome.

References

Barber, K. S., and Martin, C. E. 1999. Agent autonomy: Specification, measurement, and dynamic adjustment. In *Proceedings of the Autonomy Control Software Workshop at Autonomous Agents*.

Bonasso, P. 1999. Issues in providing adjustable autonomy in the 3T architecture. In *Proceedings of the AAAI Spring Symposium on Agents with Adjustable Autonomy*.

Cesta, A.; D'Aloisi, D.; and Collia, M. 1999. Adjusting autonomy of agent systems. In *Proceedings of the AAAI Spring Symposium on Agents with Adjustable Autonomy*.

Dignum, F.; Morley, D.; Sonenberg, E. A.; and Cavedon, L. 2000. Towards socially sophisticated BDI agents. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS'2000)*.

Ferguson, G., and Allen, J. 1998. TRIPS: Towards a mixed-initiative planning assistant. In *Proceedings of the AIPS Workshop on Interactive and Collaborative Planning*.

Fong, T.; Thorpe, C.; and Baur, C. 1999. Collaborative control: A robot-centric model for vehicle transportation. In *Proceedings of the AAAI Spring Symposium on Agents with Adjustable Autonomy*.

Georgeff, M. P., and Ingrand, F. F. 1989. Decision-making in an embedded reasoning system. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*.

Lupu, E., and Sloman, M. 1999. Conflicts in policy-based distributed systems. *IEEE Transactions on Software Engineering, Special Issue on Inconsistency Management* 25(6).

Moffett, J. D., and Sloman, M. S. 1993. Policy hierarchies for distributed systems management. *IEEE Journal on Selected Areas in Communications* 11(9).

Myers, K. L. 1996. Strategic advice for hierarchical planners. In Aiello, L. C.; Doyle, J.; and Shapiro, S. C., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifth International Conference (KR '96)*. Morgan Kaufmann Publishers.

Myers, K. L. 2000. Domain metatheories: Enabling user-centric planning. In *Proceedings of the AAAI Workshop on Representational Issues for Real-World Planning Systems*.

Pynadath, D. et al. 1999. Electric Elves: Applying agent technology to support human organizations. In *Proceedings of the AAAI Fall Symposium on Socially Intelligent Agents – The Human in the Loop*.

Rao, A., and Georgeff, M. 1995. BDI agents: From theory to practice. In *Proceedings of the International Conference on Multi-Agent Systems (ICMAS-95)*.

Wilkins, D. E., and Myers, K. L. 1995. A common knowledge representation for plan generation and reactive execution. *Journal of Logic and Computation* 5(6).