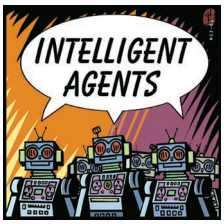
Scaling Up Agent Coordination Strategies



The value of an intelligent agent coordination strategy lies in how well it scales along various dimensions of stress. Understanding the agent population, its task environment, and expectations about its collective behavior are central to mapping the space of potential approaches.

Edmund H. Durfee
University of Michigan

Deploying intelligent agents to do peoples' bidding in environments ranging from Internet marketplaces to Mars has recently received much attention. Exactly what an agent is and in what sense a computational agent can behave intelligently remain the subject of considerable debate. However, most would agree that coordination—an agent's fundamental capability to decide on its own actions in the context of the activities of other agents around it—is a central concern of intelligent agency. Without coordination, agents can unintentionally waste their efforts and squander resources or fail to accomplish objectives that require collective effort.

Advances in agent-oriented software engineering make it possible to develop complex, distributed systems, but the component agents must be able to act and interact flexibly.[1] Choosing an effective coordination strategy requires identifying the dimensions in which the strategy needs to scale and determining how well it responds to being stressed along those dimensions. Characterizing agent population properties, their task environments, and their collective behavior is key to understanding the capabilities and limitations of coordination strategies that support flexible component agent interaction.

AGENT COORDINATION

With few exceptions, an agent dispatched to an environment will likely share it with other agents. Some proposed strategies for planetary exploration, for example, involve sending a team of robots. Thus, a fundamental agent requirement is the ability to coordinate its own actions with those of other agents.

Coordination does not imply cooperation: An effective competitor will coordinate decisions to maximize its advantage against an opponent, such as a company timing a product promotion to undercut a rival. Nor does it imply reciprocation: An agent can coordinate its actions with another agent unaware of its presence, as when an automobile driver passes a second driver whose mind is entirely elsewhere.

Not surprisingly, various coordination strategies for computational agents have emerged over the years. It does not seem possible, however, to devise a coordination strategy that works well under all circumstances; if such a strategy existed, human societies would substitute it for the myriad constructs employed today such as corporations, governments, markets, teams, committees, professional societies, and mailing groups. Whatever strategy we adopt, certain situations can stress it to the breaking point.

Any proposed coordination strategy must therefore address how it scales to increasingly stressful situations. To map the space of potential coordination strategies, we must therefore identify important dimensions along which they must scale and then evaluate their responses to stresses along those dimensions.

DIMENSIONS OF COORDINATION STRESS

An agent coordination strategy must consider the agent population, task environment, and solution. For each of these properties, we consider three of many important dimensions; although these dimensions are not necessarily orthogonal, treating them as such helps characterize the coordination challenges.

0018-9162/01/$10.00 © 2001 IEEE

Distributivity **increases uncertainty about which agents are currently sharing the task environment and what each agent is or should be doing.**

Agent population properties

The most obvious properties that impact coordination are those of the agent population. Certainly, one challenge in scaling any coordination strategy is handling more agents. Coordination strategies that rely, for example, on a centralized coordinator to direct the efforts of other agents can quickly degrade as the coordinator becomes incapable of processing the interactions of an increasing number of potentially interacting agents. Other important agent properties include heterogeneity and complexity.

Quantity. If each agent interacts with every other agent, the number of paired interactions will grow quadratically with the number of agents. Because interactions often occur within the context of ever-larger groups of agents, not just pairs, the coordination problem can increase exponentially: If every agent could choose among b actions, each potentially having a different impact on other agents, the space of all possible action combinations would be b^n for n agents. Even if the coordination search problem was divided equally among the n agents, a problem one-nth of b^n will still exceed an agent's computational limitations as n grows.

Heterogeneity. In addition to having different goals, beliefs, or expertise, agents also can have various communication languages, ontologies, or internal architectures. Whether a coordination strategy scales to increasingly heterogeneous populations depends on the degree to which it expects or enables agents to communicate with, share their abilities with, and agree with one another.

Complexity. In terms of agent properties, *complexity* refers to how hard it is to predict what an inherently versatile agent will do. One characteristic of an intelligent agent is the ability to flexibly decide for itself what goals to pursue at a given time and how to pursue them. Coordinating with less complex agents that single-mindedly perform a specialized task is easier because they are more predictable. Coupling their complexity with the possibility that inherently versatile agents will have overlapping spheres of interest and ability can put enormous stress on an agent coordination strategy.

Task-environment properties

The environment in which agents operate and the tasks they are expected to accomplish within it are major considerations in developing or choosing a coordination strategy. Real task environments often introduce complications that blur the understanding of a coordination strategy. For example, differences in performance might be due to the quality of knowledge given to individual agents rather than to the efficacy of the coordination strategy. For this reason, researchers often use idealized task domains such as the Prisoner's Dilemma, distributed sensor networks and transport problems.[2] As the "Transport Problems" sidebar indicates, even in an abstract task environment, there are numerous possible dimensions for scaling the difficulty of coordination.

Degree of interaction. The environment or task can lead to interactions that materially impact the agents. Coordination requires exerting some control over interactions, and greater interaction implies the need for more coordination. Suppose that an interaction involves an issue that concerns more than one agent—for example, who gets to use a resource, the status of some feature of the world, or who is supposed to do what task. The degree of interaction increases with the number of agents concerned with the same issues and as more issues become a concern to each agent. Consequently, settling some issues commits agents to interactions that in turn impact how they should settle other issues. As the web of dependencies grows, coordination strategies can have difficulty scaling.

Dynamics. A multiagent setting in which different agents monitor only portions of the environment, and each agent can change its mind about what goals to pursue and how to pursue them, complicates the dynamics of coping with environmental changes. In more static task environments, agents are more likely to converge on coordinated agreements, but coordination strategies that scale to highly dynamic task environments are relatively uncommon because the solutions they generate cannot keep up with changes in the task environment.

Distributivity. In some task environments, the agents are conceptually collected together and tasks originate at one point. In other task environments, the agents are highly distributed and tasks are inherently distributed among them. *Distributivity* stresses a coordination strategy because it increases uncertainty about which agents are currently sharing the task environment and what, if anything, each agent is or should be doing.

Solution properties

Coordination strategies addressing scaling issues must still yield satisfactory solutions, and we can make solution criteria themselves more stringent along dimensions that include quality, robustness, and overhead limitations.

Quality. We can measure a solution's quality in terms of how well it coordinates agent interactions or how efficient it is in using agent resources and abilities to settle issues. Higher quality can correspond to near-optimal coordination, while lower quality might correspond to merely achieving a satisfactory level of coordination. In some cases, simply avoiding dis-

agreement or conflict is good enough. A coordination strategy for an automobile intersection, for example, could specify requirements ranging from simple crash prevention to ensuring that drivers' wait times do not exceed a certain upper limit. Increasing demands puts greater stress on the coordination strategy.

Robustness. Uncertainty or other task environment dynamics can affect a solution's robustness. For example, if a coordination strategy cannot keep up with a particularly dynamic task environment, it can become outdated. The coordination strategy should anticipate, either implicitly or explicitly, the range of conditions under which the solution it provides will be followed. In a task environment in which a minor deviation from expectations can lead to severe consequences, finding assured robust solutions can be imperative.

Overhead limitations. The coordination strategy's costs could include computation requirements, communication overhead, time spent, and so on. If, for example, communication is costly and time consum-

Transport Problems

In their simplest forms, transport problems involve agents moving one or more entities from one location to another along connecting paths. Examples include message routing, cargo delivery, and evacuation tasks. Describing and visualizing transport problems, which are commonly used in agent coordination research, is straightforward. However, these problems can be sufficiently complex to make coordination difficult.[1]

When transport agents can make decisions independently, scaling up to large numbers is easy; challenges lie in other dimensions of stress. In the evacuation task shown in Figure A, for example, limiting a transport's capacity or prohibiting more than one transport from being in a location or traversing a path at one time can increase the degree of interaction. Imposing more stringent efficiency demands—for example, not just getting evacuees to safety but getting them there as quickly as possible or with minimal movement—likewise increases stress.

Other dimensions complicating coordination in transport problems include

- distributivity—only agents near evacuees, rather than a central dispatcher, know about particular transport tasks;
- dynamics—new evacuees randomly appear over time, or locations and connections appear and disappear unpredictably; and
- heterogeneity—only some transports can move particular evacuees.

If minimizing bandwidth is a solution requirement, avoiding duplicate efforts or cooperatively helping others find safe locations is harder and could involve such insect-inspired techniques[2] as marking paths to either repulse transports from already visited locations or attract others to safe locations.

References

1. K. Fischer, J.P. Muller, and M. Pischel, "AgenDA—A General Testbed for Distributed Artificial Intelligence Applications," *Foundations of Distributed Artificial Intelligence*, G.M.P. O'Hare and N.R. Jennings, eds., John Wiley & Sons, New York, 1996, Chapter 15.
2. V. Parunak, "'Go to the Ant': Engineering Principles from Natural Agent Systems," *Annals Operations Research*, vol. 75, 1997, pp. 69-101.

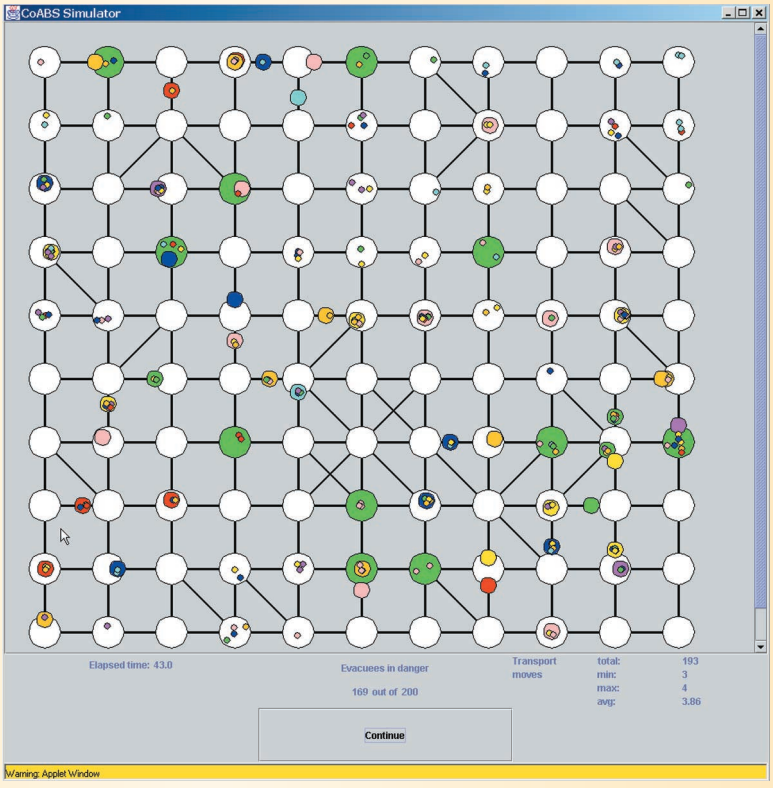


Figure A. Simulation of an evacuation task. Locations (large circles) are connected in a grid as well as by random diagonal links. Transports (large colored circles) move among locations, pick up evacuees (small colored dots), and move them to safe (green) locations.

Heterogeneity is desirable because it increases the systemwide capabilities, allowing agents with complementary attributes to combine their efforts to achieve objectives.

ing, a coordination strategy might need to reduce its demands for information exchange among agents; beyond some point, it will have to make high-quality coordination decisions lacking information it would otherwise expect to have. Questions can therefore arise about whether a coordination strategy can scale well to environments that impose more stringent limits on the costs the strategy incurs.

Combining dimensions

Scaling along combinations of these various dimensions obviously poses even greater challenges. Handling complex agents is much harder, for example, if they are complex in heterogeneous ways. The inherent delays associated with propagating changes in a high-distributivity setting likewise compound the difficulties that arise in a dynamic task environment.

Coordination strategies therefore tend to make assumptions about which dimensions a given application domain is likely to stress. A strategy for handling complex, heterogeneous agents might require limiting the number of agents. Lowering agent interaction can simplify distributivity in a dynamic task environment by localizing the need to propagate awareness. As long as minimizing costs and delays is not a major objective, continually monitoring and updating a coordination solution can improve robustness without sacrificing quality.

CHARACTERIZING COORDINATION STRATEGIES

Considering qualitative "low" and "high" values for the three dimensions of the three major types of properties—population, task environment, and solution—that could be scaled to make coordination harder would result in 512 combinations requiring a variety of coordination strategies. For purposes of this discussion, however, the following examples provide just a selection of coordination strategies and the kinds of scaling for which they are particularly well suited.

More agents

To some, scaling up means being able to handle more agents, and handling more agents is usually harder than handling fewer. Trying to get a large population of complicated, self-interested, interacting agents to behave efficiently and robustly in a dynamic environment is a difficult task. Usually, something must give: Coordination strategies that scale well to numerous agents tend to deal poorly with other confounding dimensions.

For example, cellular automata often deal with large numbers of entities that typically use rules to react to their local environment—such as deactivating when too few neighbors are active or activating when enough neighbors are active. These simple local changes can cause activity patterns to emerge in the population. Physics-based models of large computational agent ecosystems can even lead to designs of metamorphic robots made up of many small pieces that shift and flow to adapt to the environment.[3] Similarly, systems based on insect metaphors assume that each agent is a relatively simple automaton and that local interactions cause emergent properties to arise.[4]

These strategies assume little complexity or heterogeneity in the agent population, focus on limited interactions, and are often satisfied with statistical system performance rather than being concerned about using each agent efficiently or making optimal choices.

Successfully scaling up to numerous agents generally requires that each agent only needs to interact with a predetermined number of other agents based on features such as their physical locations or their tasks. Thus, many mobile agents can be dispersed to perform information-gathering tasks independently, interacting only indirectly to contend for bandwidth or server cycles.[5] Similarly, large-scale coalition formation can be an emergent process involving incrementally growing groups of agents that encounter one another and discover advantages of banding together.

More heterogeneity

Heterogeneity can help in scaling up large agent populations if disparate agents do not need to interact. However, heterogeneity typically is desirable in a system because it increases the systemwide capabilities, allowing agents with complementary attributes to combine their efforts to achieve objectives beyond what they can achieve individually. Once the agent population is no longer homogeneous, the agents must be able to understand and describe what they can do and find other agents with which to work.[6] To scale along the heterogeneity dimension, coordination strategies need to support the ability of agents to describe themselves and to find one another, such as hardwiring implicit acquaintanceships among agents.

The contract net protocol[2] and its descendants have been a mainstay coordination strategy for handling heterogeneity. In this protocol, agents dynamically assign tasks to other available agents that are capable of doing the tasks. In its simplest form, an agent broadcasts an announcement of the task along with criteria the other agents can use to decide whether they are eligible to take on the task and, if so, what information to supply in making a bid for the task. The agent with the task can choose from among the responses to make an assignment.

The contract net protocol scales well to an open system of heterogeneous agents, but the broadcast communication requirements can become problematic as the number of agents increases. One solution is to

maintain a more centralized registry of agents and their capabilities to use in discovering promising matches. Strategies that support agent registration and matchmaking,[7] such as Sun Microsystems' Jini networking technology (http://www.sun.com/jini), allow agents to find one another by describing the services they need or provide.

More generally, formalisms for communicative acts, such as the Foundation for Intelligent Physical Agents (http://www.fipa.org), permit a broad array of conversation policies in support of flexible agent interactions among heterogeneous agents. Many of these concepts are converging in comprehensive frameworks that support heterogeneous agent-based systems, such as the Defense Advanced Research Projects Agency's Control of Agent-Based Systems (CoABS) Grid (http://coabs.globalinfotek.com).

More complexity

Heterogeneity emphasizes the challenges that accrue when specialist agents need to identify one another and team to provide broader services. Additional complications arise when agents are individually more complex, which typically means that each agent is more versatile. Each agent must then decide which of the many possible roles it should play, and it must consider the potential alternative activities of other agents.

Scaling up to more complex agents means that teaming involves not only finding an available agent with appropriate capabilities, but also selecting the agent whose other talents are least in demand by other teams. Instead of localized agent interactions within smaller teams, their partial substitutability for one another leads to complex chains of agent dependencies: How some teams are formed can affect other teams' desirability. Agents therefore must be increasingly aware of the agent network's broader needs.

Even when agents do not need to team up but merely must coexist and stay out of each other's way, each agent's increased versatility makes anticipating what others will do much more difficult. Strategies for increasing awareness about other agents' planned activities is paramount because being prepared for anything that another agent could choose to do might be impossible. These strategies can include statistical learning, observing agents to infer their current plans, or communicating information agents can use to model one another's intentions.

For example, in a distributed constraint satisfaction process, agents can converge on mutually compatible plans to accomplish their objectives in several ways. In this process, tentative plan choices are propagated among agents; when inconsistencies are detected among the choices of a subset of agents, some of the agents perform systematic backtracking. Strategies for making this process more efficient include parallel asynchronous exploration and dynamically reordering which agents should try other alternatives by identifying the constraints that are the most difficult to satisfy.[2]

Higher degree of interaction

As the number and complexity of agent interactions grow, coordination becomes intractable. Reducing or eliminating interactions is an effective means of addressing coordination. When agents are only concerned about interactions with a small number of local neighbors, scaling to large numbers of agents is easier. Thus, localizing interactions can obviate the need for more complicated coordination strategies.

One often-used technique for controlling the degree of interaction is to impose a relatively static organizational structure on agents. In this structure, each agent has a role to play in the organization, including its own sphere of control and knowledge of agents playing related roles. Giving each agent the resources it requires to fulfill its role eliminates the need for resource negotiation, and providing knowledge of other agents' roles identifies the agents that need to communicate and about what. Such a structure simplifies coordination and allows larger, more complex agent systems to succeed in more demanding task domains. The challenge, of course, is designing organizations that match the agent population and the task environment's needs.[8]

Some multiagent tasks require tight interactions among agents. Examples include combat flight operations[9] and pursuit tasks in which predators need to surround a prey. As the "Pursuit Problems" sidebar describes, interactions are not a side effect of such applications but rather their specific purpose. Therefore, an emphasis on agent teams is appropriate, leading to frameworks in which system designers explicitly describe team behavior, with particular attention to which team members should interact, when, and how.[9,10]

When agents must formulate compatible plans but no previous examples are available, they need techniques for reasoning about how agent actions can facilitate, hinder, or even disable others' actions. Merging individually formulated plans that permit agents to successfully accomplish their activities without interfering with one another can be a useful technique.[11,12]

More dynamic interaction

Whether viewed as a population of individuals or as a team, a multiagent system that operates in a dynamic task environment must contend with changes in plans, goals, and conditions. Tasks that agents could previously carry out independently might now require

Teaming involves finding an available agent with appropriate capabilities and selecting the agent whose other talents are least in demand by other teams.

Pursuit Problems

In contrast to transport problems, in which agents may be able to separately accomplish different goals, pursuit problems require two or more agents to cooperate to achieve a shared goal—for example, predators surrounding and capturing one or more prey.[1] As Figure B shows, pursuit tasks are usually simulated on a two-dimensional grid, and each agent can only move to a vacant contiguous square. Because many real-life problems require agent cooperation, researchers have stressed the pursuit task along various dimensions to discover different coordination strategies' limitations.

Expecting more from a solution while limiting the resources available is one obvious way to make pursuit harder—for example, requiring that predators catch the prey every time, as quickly as possible, or with little or no communication. Limiting each predator's perceptual range to prevent it from locating the other agents, possibly including the prey—a form of inherent distributivity—can likewise be problematic. Increasing the numbers of predators and prey makes it more difficult to determine which combination of predators should pursue which prey. If predators and prey can appear and disappear unexpectedly, the challenge is to fashion a robust strategy responsive to such changes.

An extreme case of the problem involves mutual pursuit, in which members of one predator group try to surround another group's member who, with its mates' help, tries to turn the tables on its would-be captors. The uncertainty of an agent as to whether it is predator or prey at any given time stresses the complexity dimension.

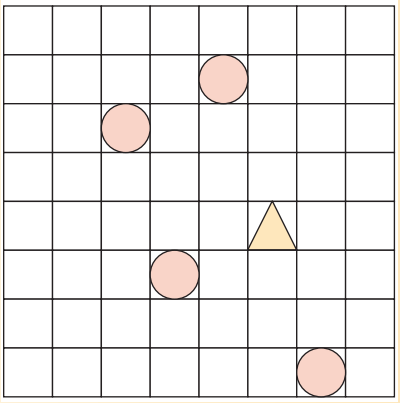


Figure B. Simulation of a pursuit task. Predators (circles) that individually attempt to capture a prey (triangle) can easily fail. If each predator simply attacks the side of the prey closest to it, one or more other sides may remain open. Unless they surround the prey before closing in, predators that get next to the prey first can chase it to a side or corner of the grid, resulting in a stalemate.

Reference

1. L. Gasser et al., "Representing and Using Organizational Knowledge in Distributed AI Systems," *Distributed Artificial Intelligence Vol. 2,* M.N. Kuhns and L. Gasser, eds., Pitman, London, 1989, pp. 55-78.

interaction—for example, when a resource becomes unusable, leading to contention for remaining resources. Existing agreements might be subject to review as some team members change their priorities or recognize that their individual intentions, or those of the team, are no longer relevant in the new context.

Various conventions can specify what agents should do when they begin to question their commitments because of task-environment dynamics.[13] One such convention seeks to ignore dynamics entirely by insisting that agents must fulfill their commitments regardless of changing circumstances. Other alternatives include allowing agents to renege on commitments if they pay a penalty or allowing agents to abandon obsolete commitments if they notify team members, thus potentially stimulating the formation of different commitments.

In fact, dynamic task environments can suggest that agents should never view plans as being anything more than tentative. Agents could unilaterally change their plans and begin acting on new plans before reaching agreement across the team. This approach has the potential for leading to inefficient collective activities because of information delays, causing chain reactions and even race conditions among changes. However, under some limiting assumptions about how and when agents can make unilateral changes, iterative coordination and execution techniques can lead to flexible coordinated behavior in dynamic task environments.[2]

More distributed interaction

Even when agent interactions are few and not undergoing dynamic changes, a task environment can stress agents if interactions requiring coordination are hard to anticipate. If agents are acting based on privately held information about goals and methods, identifying which agents might interact is particularly difficult.

One response is to anticipate all potential actions agents might take and impose restrictions that prohibit undesirable interactions. Such "social laws" ensure that law-abiding agents need not worry about undesirable interactions, no matter what goals and plans they adopt.[14] In human terms, as long as all drivers obey traffic laws, for example, they should be able to eventually reach their destinations without colliding.

Another option is to help potentially interacting agents efficiently find one another. Auctions, for example, bring together related buyers and sellers.[2] Creating agents that represent resources over which agents might contend helps identify resource demands. Interaction helps agents discover other agents with which they can form persistent aggregations.

Without an identifiable context for aggregation, however, agents must somehow test for possible interactions against all other agents. To accomplish this, a

coordinator could collect summary information on all relevant agents and use its global awareness to inform agents about potential interactions.[12] Alternatively, each agent could broadcast information to all other agents so they all have sufficient awareness of the global picture. These iterative exchanges help the overall system cooperatively achieve its objectives.

Nearer to optimality

Optimal coordination is generally desirable but rarely feasible because it generally requires substantial computation and communication overhead. In a dynamic task environment or an environment with many agents or complex interactions, a locally optimal solution is often acceptable. Often, simply finding a coordinated solution that does well enough—one that avoids conflicts among agents or ensures eventually achieving goals—is enough.

In market-based methods for finding an optimal solution, agents maximize resource allocation efficiency by using iterated auction bidding rounds to balance supply and demand.[2] Active research is extending these coordination strategies to scale them along other dimensions so they can handle larger numbers of agents as well as agents with higher degrees of interaction and greater dynamics.[15] Distributed, rational decision-making methods[2] based on multiagent extensions of Markov decision processes[16] can find an optimal policy based on a particular coordination protocol employed at runtime—for example, to increase agents' awareness of the global situation.

More robustness

An optimal solution can break down when the world deviates from the coordination strategy's underlying assumptions. Whether a coordination strategy can scale to domains where robust performance is difficult but necessary can thus become important.

Building a solution that gives agents sufficient flexibility to work around new circumstances within their original coordination agreement can increase robustness. For example, building slack time into scheduled activities or not committing to details can leave agents with more maneuvering room when things do not go according to plan. Typically, more robust coordination decisions are less efficient because they reserve resources and options for fallback contingencies, suboptimally assigning tasks among agents, a feature typical of stable organizations.[2,8]

Alternatively, a coordination strategy could monitor its solution's execution and make repairs as needed. An example of this approach is a teamwork model that includes a convention for responding when continued pursuit of a joint commitment is senseless.[17] In some cases, developing generic monitoring and recovery methods for the coordination processes might be possible.[18]

Lower overheads

Reducing coordination strategy overhead is important in application domains with limited communication channels and minimal computational resources. As bandwidth becomes more limited, for example, agents must make coordination decisions without exchanging enough information to maintain an expected level of global awareness.

To maximize resource allocation efficiency, agents can use iterated auction bidding rounds to balance supply and demand.

Time-constrained coordination solutions involving the iterative exchange of increasingly detailed information about agents' plans and intentions can limit communication and computation overhead at the expense of quality.[12] Alternatively, agents can avoid a coordination chain reaction by tolerating outdated decisions. When communication is at a premium or even impossible, using observations to model others or reasoning to converge on decisions is another alternative.

Sometimes coordination resources are sporadically available. In some coordination scenarios, agents can use plentiful resources to build more complete models of their roles and contingent plans, which they can exploit when they move into situations in which further communication and computation are unsafe or infeasible.[19,20]

A variety of promising ideas exist for designing computationally realizable agent coordination strategies that work well under a broad range of circumstances. Most coordination strategies can scale along multiple dimensions, but each has its limits. Researchers face the challenge of developing a better—preferably quantifiable—understanding of exactly how far different coordination strategies can scale along the dimensions required to apply intelligent agent systems to increasingly challenging problems. ✸

Acknowledgments

Many ideas in this article arose through interactions with researchers in the CoABS project, including Craig Boutilier, Jim Hendler, Mike Huhns, David Kotz, James Lawton, Onn Shehory, Katia Sycara, Milind Tambe, and Sankar Virdhagriswaran. Milind Tambe provided valuable feedback on an early draft. DARPA, in part, supported this work through Rome Labs (F30602-98-2-0142).

References

1. N.R. Jennings, "An Agent-Based Approach for Building Complex Software Systems," *Comm. ACM*, Apr. 2001, pp. 35-41.
2. G. Weiss, ed., *Multiagent Systems: A Modern Approach*

to Distributed Artificial Intelligence, MIT Press, Cambridge, Mass., 1999.
3. H. Bojinov, A. Casal, and T. Hogg, "Multiagent Control of Self-Reconfigurable Robots," *Proc. 4th Int'l Conf. Multiagent Systems* (ICMAS 2000), IEEE CS Press, Los Alamitos, Calif., July 2000, pp. 143-150.
4. J. Ferber, *Multiagent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley, Harlow, UK, 1999.
5. R.S. Gray et al., "Mobile-Agent versus Client/Server Performance: Scalability in an Information-Retrieval Task," tech. report TR2001-386, Dept. of Computer Science, Dartmouth College, Hanover, N.H., Jan. 2001.
6. T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, May 2001, pp. 34-43.
7. K. Decker, K. Sycara, and M. Williamson, "Middle-Agents for the Internet," *Proc. 15th Int'l Joint Conf. Artificial Intelligence* (IJCAI 97), Morgan Kaufmann, San Francisco, 1997, pp. 578-583.
8. M.J. Prietula, K.M. Carley, and L. Gasser, eds., *Simulating Organizations: Computational Models of Institutions and Groups*, AAAI Press/MIT Press, Menlo Park, Calif., 1998.
9. M. Tambe and W. Zhang, "Towards Flexible Teamwork in Persistent Teams: Extended Report," *J. Autonomous Agent Multiagent Systems*, June 2000, pp. 159-183.
10. B.J. Grosz and S. Kraus, "Collaborative Plans for Complex Group Action," *Artificial Intelligence*, vol. 86, no. 2, 1996, pp. 269-357.
11. E. Ephrati, M.E. Pollack, and J.S. Rosenschein, "A Tractable Heuristic That Maximizes Global Utility through Local Plan Combination," *Proc. 1st Int'l Conf. Multiagent Systems* (ICMAS 95), IEEE CS Press, Los Alamitos, Calif., 1995, pp. 94-101.
12. B.J. Clement and E.H. Durfee, "Top-Down Search for Coordinating the Hierarchical Plans of Multiple Agents," *Proc. 3rd Int'l Conf. Autonomous Agents*, ACM Press, New York, May 1999, pp. 252-259.
13. N.R. Jennings, "Commitments and Conventions: The Foundation of Coordination in Multiagent Systems," *Knowledge Eng. Rev.*, vol. 2, no. 3, 1993, pp. 223-250.
14. Y. Shoham and M. Tennenholtz, "On Social Laws for Artificial Agent Societies: Off-line Design," *Artificial Intelligence*, vol. 72, no. 1-2, 1994, pp. 231-252.
15. Y. Fujishima, K. Leyton-Brown, and Y. Shoham, "Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches," *16th Int'l Joint Conf. Artificial Intelligence* (IJCAI 99), Morgan Kaufmann, San Francisco, 1999, pp. 548-553.
16. C. Boutilier, "Multiagent Systems: Challenges and Opportunities for Decision-Theoretic Planning," *AI Magazine*, Winter 1999, pp. 35-43.
17. S. Kumar, P.R. Cohen, and H.J. Levesque, "The Adaptive Agent Architecture: Achieving Fault-Tolerance Using Persistent Broker Teams," *Proc. 4th Int'l Conf. Multiagent Systems* (ICMAS 2000), IEEE CS Press, Los Alamitos, Calif., 2000, pp. 159-166.
18. C. Dellarocas and M. Klein, "An Experimental Evaluation of Domain-Independent Fault Handling Services in Open Multiagent Systems," *Proc. 4th Int'l Conf. Multiagent Systems* (ICMAS 2000), IEEE CS Press, Los Alamitos, Calif., 2000, pp. 39-46.
19. E.H. Durfee, "Distributed Continual Planning for Unmanned Ground Vehicle Teams," *AI Magazine*, Winter 1999, pp. 55-61.
20. P. Stone and M. Veloso, "Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork," *Artificial Intelligence*, vol. 110, no. 2, 1999, pp. 241-273.

Edmund H. Durfee *is a professor of computer science and engineering at the University of Michigan, Ann Arbor. His research interests include multiagent systems, real-time intelligent control, and cooperative problem solving for applications ranging from interacting unmanned air and ground vehicles to supporting human collaboration. Durfee received a PhD in computer science from the University of Massachusetts. He is member of the ACM and the AAAI and a senior member of the IEEE. Contact him at durfee@umich.edu.*