article 32

Artificial Intelligence in Organization and Management Theory

Models of Distributed Activity

Edited by

Michael MASUCH

and

Massimo WARGLIEN

Center for Computer Science
in Organization and Management (CCSOM)
University of Amsterdam
Amsterdam, The Netherlands

N·H

1992

NORTH-HOLLAND
AMSTERDAM • LONDON • NEW-YORK • TOKYO

CHAPTER 4
PLURAL-SOAR: A PROLEGOMENON TO ARTIFICIAL AGENTS AND ORGANIZATIONAL BEHAVIOR[15]

Kathleen Carley
Johan Kjaer-Hansen
Allen Newell
Michael Prietula

1. Introduction

In this chapter, we show how artificial intelligence (AI) can be applied to the study of organizations. Specifically, we construct an AI model of a small organization in which intelligent agents communicate and cooperate to accomplish a task. The task involves filling orders by retrieving items stored at various locations in a warehouse. Each agent is represented on its own computer using a sophisticated software architecture (called Soar) that is capable of serving as a basis for general intelligence and learning from experience. To frame this research we focus on this question: do the communication and memory capabilities of the agent affect

[15] This research was supported in part by the Defense Advanced Research Projects Agency (DOD), and monitored by the Avionics laboratory, Air Force Wright Aeronautical Laboratories, Aeronautical Systems division (AFSC), Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-87-C-1499, ARPA Order No. 4976, Amendment 20. In addition, support was provided by the Fulbright Commission, The Danish Research Academy under a shared grant V890138, the National Science Foundation under grant SES-8707005 and the Center for the Management of Technology, Graduate School of Industrial Administration, Carnegie Mellon University.
The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, the National Science Foundation, the U.S. Government, the Fulbright Commission, or the Danish Government.

which coordination scheme is most effective? We find that increasing agents' capabilities can, under certain conditions, actually degrade organizational performance.

1.1 Plural-Soar: a prolegomenon to artificial agents and organizational behavior

Organizations are comprised of intelligent people who have a native capability to learn and solve problems unrelated to their organizational role. Organizations direct individuals' specific problem-solving behavior by providing them with certain tasks, and hence, goals to achieve within those organizational roles. Further, organizations restrict what capabilities an individual should or can employ by limiting the number of "acceptable" behaviors in any task related situation either through (1) establishing a formal or informal norm of performance within the role, or (2) inhibiting the requisite support to apply those behaviors effectively. As a consequence of such restrictions, the match between skills and job requirements within a particular organizational structure and coordination scheme may be insufficient to achieve a maximally efficient organization.

Examining the three-way relationship between individual's skills, job requirements, and the schemes for coordinating individuals within the organization requires a micro-view of the organization. Such micro-level analyses have traditionally been carried out using experimental subjects within the laboratory or through in-depth case analyses of particular organizational decisions. Both techniques have limitations as well as merits. While it is not our objective to critique these techniques, we do note that neither facilitates systematically examining the three-way relationship with minimal cost. For example, subjects, whether within the lab or the actual workplace, bring with them a set of preconceived ideas or preferences for doing certain tasks in certain ways; they also have certain skills that are difficult to control, let alone eliminate. Artificial agents, however, need suffer no such handicap. Indeed, such agents can be "built" to the specification needed for the experiment. Consequently, it should be possible to develop a micro-level theory of organizations using artificial agents.

Such artificial agents, however, are often designed to do a single specific task, such as playing chess. Yet, in human organizations the agents (human beings) are capable of doing, and do perform, a wide variety of tasks. This suggests that to develop micro-level organizational theories we will need artificial agents with general intelligence. Our understanding of individual cognition now permits extensions beyond specific models of a particular task, such as chess machines, to models of general intelligence. Such models can be used to replicate fundamental methods and deliberation procedures independent of specific task requirements. One such candidate is Soar (Laird, Newell, and Rosenbloom, 1987; Laird, Rosenbloom, and Newell, 1986a; Laird, Rosenbloom, and Newell, 1986b). Using models such as Soar, it should now be possible to develop micro-level theories of organizational behavior, which is our ultimate goal.

In this chapter, however, our aim is more modest and we simply present a technique for examining at the micro-level the three-way relationship between individuals' skills, job requirements, and the coordination scheme. This technique uses a set of artificial agents each realized on its own machine, in an organization, engaged in a distributed task. We refer to the resulting system as Plural-Soar. We will show that Plural-Soar can be used to investigate the joint impact of organizational coordination and people-skills on organizational efficiency. This chapter is a first step in the development of a truly cognitive social theory of organizational behavior. Our goal in this chapter is to report on the viability of using an organization of independently functioning artificial social agents, which are generally intelligent and not specifically designed for the task in which they are engaged, to perform a distributed decision-making task. Our purpose here is to demonstrate that artificial cognitive agents make micro-level social theory possible, not to make theoretical predictions.

We use the *Warehouse Task* in which the agents must fill a set of orders sent to the warehouse by locating the requested items within the warehouse. We use this task, not because we are concerned with warehouses, or the efficient handling of orders, but because it is a readily understood and easily modelable task that nevertheless contains sufficient complexity to examine a variety of social and group decision-making behaviors. We model each of the agents using Soar, which is an AI software architecture capable of learning from experience (Laird, Newell, and Rosenbloom, 1987). Although the agent-models are somewhat simplified, Plural-Soar demonstrates that a set of networked computers, each running a Soar model of an intelligent agent, can effectively perform a small, but potentially realistic task of retrieving items from a warehouse. We demonstrate that both degradation and improvement in organizational performance can occur as agents change their set of skills. Necessarily, we view these results as highly context-specific, so they should not be interpreted as providing guidelines to the conditions under which organizations perform in particular ways. Rather, the result should be interpreted as the first step toward researching micro-level organizational theories through the simulation of multiple intelligent agents.

To lay the groundwork for even this first step, we list in Table 1 a set of capabilities that an intelligent agent must have if it is to act as a social agent within an organization, that is, as an organizational agent. This list of capabilities is tailored to the warehouse task and should be taken as exemplary, but not as exhaustive (for a more complete discussion of the social agent see Carley, and Newell 1990). A functioning organizational agent must demonstrate six basic capabilities. The agent must be able to (1) perceive the environment and take action; (2) remember aspects of the perceived environment; (3) follow instructions; (4) analyze the task with which it is faced and determine a course of action; (5) communicate with other agents in a variety of ways; and (6) analyze the social environment with which it is faced. All these various capabilities are affected by the agent's cognitive architecture and by the agents social position (i.e., the agent's membership in the organization, who the agent knows, what groups the agent is a member of, and so forth). Communicating with other agents includes both the style and mode of communication. Analyzing the social environment would include both modeling other agents and organizations and then acting accordingly. Such actions might include acting in accordance with norms and cultural restrictions. We will use these required capabilities as a guideline for examining the comprehensiveness of the agents we examine in this chapter.

If AI models like Soar are sufficient to describe the social agent (and hence human general intelligence), they should be able to account for all related forms of goal-oriented deliberation, including group and organizational behavior. Since organizational concepts such as "coordination", "communication", and "structure" naturally arise when two or more intelligent agents interact to achieve a goal, we can expect to construct models of organizations by weaving together models of generally intelligent individual agents. This is the approach we take in this chapter. Ultimately, if this approach is successful, we will not only have a robust model of group behavior, but a strong theoretical basis for the fundamental explanation of organizational behavior.

Table 1: Required capabilities of social agent

Perception and Action
- Perceives the environment
- Physically manipulates objects
- Moves self to different locations

Memory
- Location
- People
- Task

Instruction
- Can be incomplete

Task Analysis
- Decomposes task
- Coordinates subtasks for self to do

Communication Skills
- Asks questions/Provides answers
- Gives commands/Receives commands
- Talks to a single individual/Talks to a group

Social Analysis
- Models of other agents
- Model of organization

2. Organizational theory and computer simulations

Theories of organizations abound, ranging from those derived from microeconomics and statistics, such as competitive forces (Porter, 1980), agency theory (Jensen, and Meckling, 1976), transaction cost theory (Williamson, 1975), evolutionary theory (Nelson, and Winter, 1982), or team theory (Marschak, 1955; Arrow, 1979; Radner, 1987), to those arising from more traditional group and organizational perspectives, such as group theory (Cartwright, and Zander, 1960; Shaw, 1981), information processing theory (March, and Simon, 1958; Galbraith, 1973), contingency theory (Thompson, 1967; Galbraith, 1973; Katz, and Rosenzweig, 1979; Katz, and Rosenzweig, 1972), Garbage Can theory (Cohen, March, and Olsen, 1972), to those that use economic and/or structural considerations such as interlocking directorates (Pennings, 1980), structuralism (Chandler, and Dames, 1980; Galbraith, 1973), efficiency-based (Ouchi, 1980), resource-based (Pfeffer, and Salancik, 1978), population ecology (Hannan, and Freeman, 1977;

Wholey, and Brittain, 1986), or coordination theory (Malone, 1986; 1987). Despite attempts to account for individual information-processing limits in organizational decision-making processes, such as in complexity theory (Streufert, and Swezey, 1986), none of these views is based on a fully explicated and psychological plausible model of the agents which comprise organizations – humans. Specifically, building a software program in which the agents interact according to the tenets of any of these theories would require many assumptions about how humans behave, assumptions which go beyond the theories themselves. Such assumptions might drastically alter the resultant predictions of the now enhanced theory.

AI models, specifically those of generally intelligent agents, suggest a possible path out of this dilemma: using AI simulations of intelligent agents as the basis for developing organizational theories. The role of computer simulation has a rich and important history in understanding and designing complex systems, both natural and artificial. In the past twenty years, this tactic has been at the forefront of explicating the human mind. Both AI and cognitive science have made remarkable strides in defining structures and methods that can approximate intelligent behavior. Curiously, though the use of computer simulation has played a valuable, but small, role in organizational research (e.g., Anderson, and Fischer, 1986; Axelrod, 1984; Baligh, and Burton, 1981; Burton, and Obel, 1984; Carley, 1986; 1990; Cohen, March, and Olsen 1972; Ow, Prietula, and Hsu, 1989; Kumar, Ow, and Prietula, 1990; Witte, and Zimmermann, 1986), little research by organizational theorists has actively applied artificial intelligence to modeling the most prevalent of artificial systems – organizations. Cohen (1986, p. 67) summarizes the situation:

"It may seem that this convergence [artificial intelligence and organizational design] has already occurred – especially in view of Herbert Simon's founding role in both areas of research. In fact, it has not. A closer examination reveals that the overwhelming majority of organization theory simulations are composed of FORTRAN or BASIC and are internally structured as series of equations (albeit of large and nonlinear systems)".

Such equation-based approaches to studying issues of organizational theory are appropriate when (1) the size of the organization is large enough that the behavior of its individuals is describable in detached, summary, aggregated macro-terms (e.g., production levels over a period of time for organizational learning), or when (2) there is substantial control over the dominant degrees of freedom measured (e.g., organization output and structure).

Other aspects of organizational theory, however, are more microscopic and speak directly to the individual as the units of analysis. Such aspects include socialization, small group behavior, internal coordination, team performance, norm-based behavior, and organizational culture. Efforts to study groups and their organization have been active for many years (e.g., Cohen, 1962; Cohen, et al. 1969; Shaw, 1981). Nevertheless, the methods brought to bear to study groups thus far have fallen short in an essential aspect – the modeling and prediction of collective action based on a detailed theory of individual cognition. As an old adagio goes: "groups don't make decisions – people do." Yet, current approaches to studying group phenomena virtually ignore much of what we understand about human cognition (e.g., Olson, 1989). Against this background, our goal is to generate an understanding of organizational behavior from the perspective of the fundamental units which comprise it; that is, to derive a theory of the organization from a theory of the individual. This approach is important not only for theoretical purposes, but also for the practical purpose of augmenting group problem-solving with machines.

There have already been attempts at generating theories of collectives from theories of individuals that apply AI techniques to group problem-solving tasks. Within the organization theory tradition one of the first such attempts was recently published by Masuch and LaPotin (1989). Their DoubleAISS model is a symbolic, rather than equation-based model of organizational behavior in which the decision-making agents are represented as production systems operating within problem spaces. The DoubleAISS model was used to examine and extend the contentions of garbage-can theory using a rather abstract notion of task. The model was specifically designed as a model of organizations, and was not a general model of cognition capable of performing multiple, differing tasks within an organization.

From a different perspective, researchers in distributed AI have been working on the issue of coordinating a set of cooperative agents (Reid 1979; Davis, 1981; 1982; Drazovich, 1979; Durfee, Lesser, and Corkill, 1985; Fox 1979, 1981; Lesser, and Corkill, 1981; Reed, 1981; Smith, 1980; Steeb, 1980; Thorndyke, 1981). This work, unlike that of Masuch and LaPotin, tends to employ artificial agents designed for a specific task, with the details of the task itself constraining and defining performance. But, like Masuch and LaPotin, these agents are not based on a general model of cognition. Moreover, work on distributed AI rarely takes organizational theory into account and, consequently, views groups as separably intelligent agents who coordinate themselves in fashions quite different from actual human organizations.

In summary, there is a need to explore the behavior of social agents, using a general model of cognition as the basis for collective behavior to gain generalizability, admit cumulative theory building, and consolidate theories of group behavior with theories of individual behavior. There is also a need to place a set of artificial agents with such general intellectual ability in a communication and coordination structure that more nearly approximates real human organizations. Our approach is to start with a general model of cognition, Soar, create an organizational environment with a potentially real task, and craft a set of Soar agents in this environment with the goal of performing this task, Plural-Soar. As we augment the Plural-Soar agents with the knowledge and behaviors appropriate to the task and organizational structure, we expect that noticeably social behavior should emerge (Carley, and Newell, 1990). In this way, we are developing a cognitively and socially motivated theory of organizational behavior. With respect to the work presented in this chapter, we issue the following caveat: the initial task situation and the associated agents described in this chapter are quite lean.

2. The Soar architecture[16]

Soar is a system that characterizes all symbolic goal-oriented behavior as search in problem spaces and serves as an architecture for general intelligent behavior (Laird, Newell, and Rosenbloom, 1987). In the Soar nomenclature, the world is characterized by a set of *problem spaces* with associated *states*, *operators* for reconfiguring states, *goals*, and *preferences*. We can think of problem spaces as limited knowledge domains in which there is a mental model of a specific task. This mental model is a symbolic representation of the task. Associated with this problem space are a set of operators which typically correspond to the actions that the agent can take to perform this task. These operators define completely the ways in which this domain can be altered. Further, by selecting a problem space to do a task, Soar accepts as valid the implicit representation of the task and any solution that emerges from applying the operators in this problem space.

[16] This section is drawn from a description of the Soar system contained in Prietula, Hsu, Steier, and Newell, 1990.

Soar is continuously making *decisions* which, as the primitive acts of the system, affect the order in which it applies its knowledge (and hence the search paths it follows). Decisions are used to determine which problem spaces, states, or operators are appropriate and to pursue goals specified in the *goal hierarchy* by applying operators to develop new state configurations. In Soar, the knowledge required to drive correct decisions (i.e., effective and efficient search paths) is acquired in one of two ways. First, the knowledge may be directly available in long-term memory as productions. All long-term memory knowledge in Soar is realized as productions such that, when the antecedent conditions are met, the relevant productions fire. Second, the knowledge may be indirectly available through problem resolution. Problem solving ensues, for example, when the system is trying to select an operator from a set of operators, all of which might be the "best" to apply. In the Soar architecture, the mechanism that is brought to bear on problems (regardless of the nature of the problem or where it occurs) is *subgoaling*. In essence, problems are described as *impasses*, and result in the specification of a new goal and a new problem space devoted to the resolution of the impasse. Since all goal-driven behavior in Soar is characterized in the same manner, the mechanisms embodied in the entire Soar architecture are available for the resolution of the impasse; it is simply another goal to achieve (Laird, Rosenbloom, and Newell, 1986a).

Soar operates in terms of a two-phase *decision cycle*. Each cycle starts with an elaboration phase. During this phase, the description of the current situation (i.e., the contents of the working memory) is elaborated with relevant information from Soar's long-term (production) memory. All productions that can fire, can do so in parallel. These productions can set up preferences for the objects of the goal context (problem spaces, states, and operators) or for other augmentations to working memory. Once preferences have been set, those involving non-context objects are evaluated and the resultant choices made. These choices result in items that are not context objects being added to working memory. This process of production firing, preference evaluation, and working memory update is repeated until quiescence (i.e., until no more productions are eligible to fire). Then, the second phase, the decision procedure, is entered. This interprets the preferences for context objects in the working memory. This phase can result in changes in the goal context. In the Soar architecture, *preferences* are used to describe the desirability and acceptability of the alternatives being considered for selection. If the preferences uniquely identify an object to be selected, such as a particular problem space, state, or operator, the decision is made and the decision cycle repeats, starting with another elaboration phase. However, when the preferences are incomplete or inconsistent, an impasse occurs, and the Soar architecture automatically sets up a subgoal to resolve the impasse.

Given a subgoal, Soar applies its full problem-solving capability and knowledge (as it did for the higher-level goals) to resolve the impasse that caused the subgoal. There are four types of impasses in the Soar architecture: (1) *ties*, when multiple objects are considered equally acceptable for a given goal; (2) *conflicts*, when two or more objects have mutually conflicting preferences (e.g., both object1 > object2 and object2 > object1 hold at the same time, where ">" indicates a strict preference ordering such as "worse than"); (3) *no-changes*, when the elaboration phase runs to quiescence without distinguishing preferences for any objects; and (4) *constraint failures*, when either multiple objects are required for an attribute or if an object has both a "require" and a "prohibit" preference. Note that there is a difference between a lack of knowledge to suggest an object and prohibition which reflects the result of knowledge that inhibits consideration of a particular object. When a tie or conflict and a constraint-failure occur at the same time, Soar detects only the constraint-failure.

A component of the Soar architecture is a learning capacity that is brought to bear in the resolution of impasses and which relies on a common universal mechanism – *chunking* (Laird, Rosenbloom, and Newell, 1986b). Soar learns by producing chunks of knowledge (productions) as a result of resolving impasses, that is, as a result of achieving a (sub)goal. The chunks produced by Soar within the impasse-resolution process reflect the relevant objects in working memory that caused the impasse (crafted in the antecedent portion of the generated production) and the subsequent results obtained by the subgoaling search effort (crafted in the consequent portion of the generated production). Thus the results of subgoaling are chunks that embody knowledge to reduce the search effort by permitting decisions to be made. Furthermore, to the extent that subsequent encounters generate similar impasses, these decisions will be made directly and without further deliberation via subgoaling, allowing more direct (and efficient) problem-solving. As this process is recursive, it is quite possible to generate impasses (and subgoaling) while attempting to resolve an impasse through subgoaling. Since the approach to impasse resolution is consistent throughout Soar, the system can result in dramatic reductions in problem-solving effort through the exploitation of the chunking mechanism in service of the goals it is attempting to achieve. Soar, then, is unique in that goals are created, deliberated upon, and resolved (i.e., terminated) solely by the underlying architecture. Furthermore, the chunks (i.e., knowledge) created by Soar during deliberation are subsequently always available for activation, if necessary. This, in effect, reflects the non-destructive nature of the long-term memory component of Soar. Whether productions are subsequently activated depends on the specificity of the subsequent context in which the chunks are considered.

In short, Soar is a symbolic system that casts all tasks as a collection of interacting problem spaces, where each problem space is associated with goals situated within a goal hierarchy. Within a problem space, the symbolic components representing states of the problem space are changed by the application of appropriate operators. Subgoals (and hence new problem spaces) are invoked when problem-solving cannot proceed in the current problem space as a result of impasses. The resolution of impasses (i.e., subgoal success) permits problem solving to continue and forms the basis for learning. A single learning mechanism called chunking is used universally in Soar. Control over the occurrence of chunking allows various forms of learning (including no learning) to ensue.

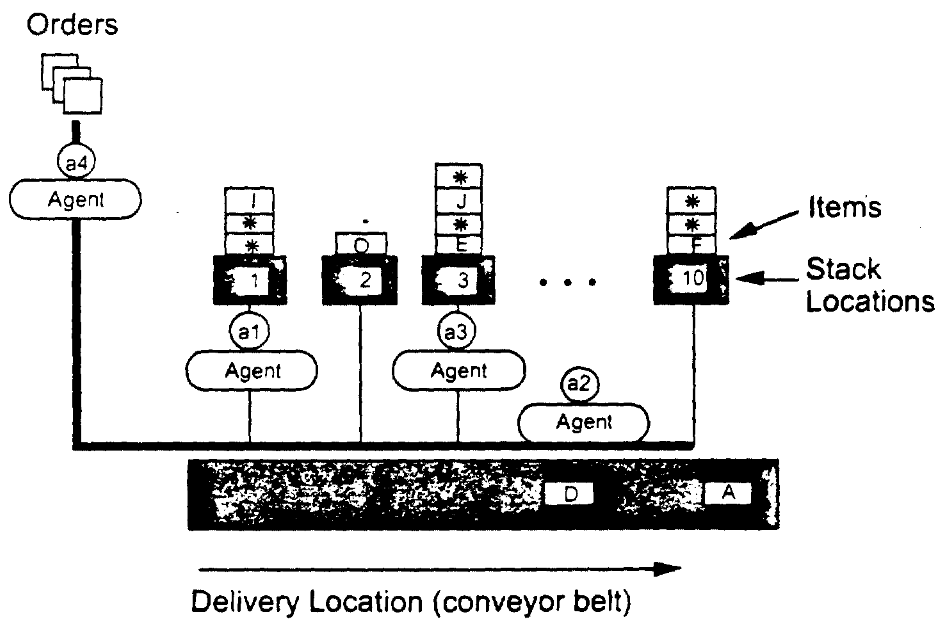
3. Plural-Soar

As noted, Plural-Soar is a system of Soar agents engaged in the warehouse task. The warehouse task is a potentially real task; that is, many organizations have warehouses containing items and orders that need to be filled. The warehouse of our experiments is a greatly simplified version of a realistic warehouse. The current warehouse environment is meant to be just complex enough to support the initial functions of the agents. As those capabilities grow, so will the richness of the warehouse.

3.1 Warehouse Task

A graphic illustration of the warehouse task is shown in Figure 1. On the floor of the warehouse is a single row of stacks. Each stack contains one or more labelled items. At a particular location in the warehouse, there is a set of posted orders. Each order contains a list of items requested by a customer, but no information about where the requested items are in the warehouse. In front of the stacks is a walkway along which agents can move, while across the walkway is a conveyor belt onto which selected orders must be placed by the agents. The basic goal associated with the warehouse task is to fill all the posted orders with the items in the warehouse. An item on an order is considered "filled" if the requested item is removed from a stack and placed on the conveyor belt. The warehouse task is thus comprised of the following entities: agents, stack locations, items, posted orders, and delivery location. The relationship among entities and permitted behaviors define constraints on the manipulable elements of the task.

Figure 1: The warehouse task



Within the warehouse an agent can only move left or right along the walkway or wait in front of a stack. If multiple agents are in front of a stack, they form a queue such that only the agent immediately in front of the stack can move items off of a stack and see the contents of the stack. While immediately in front of a stack, an agent can take items off a stack and place them on the stack to the left or right or on the conveyor belt. If the agent wants to remove an item in a stack, all items on top of it must first be moved to the left or right stacks before the required item can be extracted. Although agents in a queue cannot manipulate items in the same stack at the same time, agents immediately in front of adjacent stacks can move items to the top of their neighbor's stacks. The order list is represented as a stack to the left of all other stacks. All agents are initially located in a queue at the order stack. Each agent sequentially takes the top order off of the order stack and then attempts to fill it.

Admittedly rudimentary, the warehouse task as described does contain many of the complexities generally attributable to distributed decision-making tasks: agents work cooperatively, agents may not be engaged in face-to-face discussion, each agent has his or her own task, the organizational goal requires all agents to perform their tasks, issues of effort allocation and distributed skills arise, the task is not solved by all agents reaching consensus, and so forth. Additionally, meaningful research in coordination and communication requires surprisingly simple tasks (e.g., Carley, 1990b; Weingart, 1989). Indeed, the simplicity of the task can be advantageous, because it clarifies the relationship between organizational and individual goals and problem-solving constraints.

For the purposes of this research, the warehouse task has an important property. It affords a rich task environment which can be elaborated in ways that *realistically* represent manipulations in cognitive, as well as social, contexts. For example, manipulations may be made to agents, the task, or the design of the organization thus permitting analysis of the three-way relationship discussed in the introduction to this chapter. Agents can be manipulated by altering a variety of different factors such as: (1) the type of memory (e.g., memory of item locations, memory of order elements, of other agent's states, of other agent's goals, of other agent's knowledge); (2) effort (e.g., of traversing inter-location distances, of moving boxes); (3) risk (e.g., movement of unknown boxes, movement of heavy boxes); (4) trust (e.g., in another agent's communications); (5) ability to learn (e.g., of locations, of other agent capabilities, of strategies); and (6) ability to communicate (e.g., what is communicated to other agents – goals, locations, knowledge, as well as the costs of such communication). Order information can be manipulated by changing (1) the number of items on the order itself and (2) the number of orders an agent can peruse before selecting one to work on. Stacks can be manipulated by altering (1) the particular distribution of items over stacks relative to order demand; (2) the number of items relative to order demand; (3) the number of stacks; and

(4) the distance between stacks. Many aspects of the organizational coordination scheme can be manipulated, such as: (1) the reward structure (e.g., minimization of individual effort, minimization of communication, maximization of all agent effort, minimization of stack manipulations, response to requests of assistance from other agents); (2) organizational structures (e.g., each agent operates independently, one agent dispatches the other agents, two agents work together); and (3) the number of agents participating. The interaction of such manipulations permit complex questions to be addressed.

In this study, the particular warehouse configuration we use has the following additional features:

- The warehouse consists of 10 stacks and an order stack.
- Each stack initially contains 3 items.
- There are 15 orders.
- Each order is for a single item.
- All items are unique; that is, they occur in only one order, and only one of that item occurs in the warehouse.
- Each agent works on only one order at a time.
- There are no special purpose communication channels available such as blackboards or telephones. The only available means of communication is by broadcast.
- The warehouse is small enough that all agents in the warehouse can hear anything any other agent says.

The agents are not designed to cope exclusively with this configuration. That is, the agents we describe in this chapter can also, with no modification, operate in warehouses with different numbers of stacks, different numbers of items per stack, different numbers of orders, different numbers of items per order, and so forth. Although we shall not discuss these studies in this chapter, we have used these agents to examine other warehouse configurations.

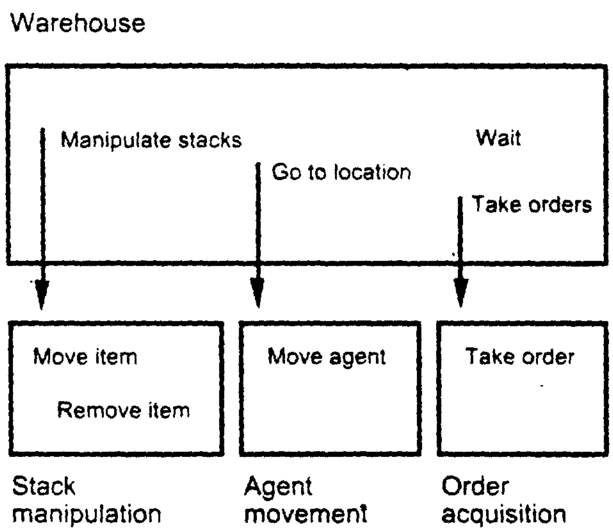
3.2 Plural-Soar agents

Each Plural-Soar agent is a Soar agent that runs on its own computer. Multiple agents working in the same warehouse act simultaneously. The actual warehouse is represented as a shared file which all agents can read and update. All agents in a single organization are identical in terms of their capabilities and what knowledge they initially bring to the task. Across organizations we vary the type of agent. In this study we will use four different types of agents: (1) the basic agent (B), (2) the basic agent plus location memory (BL), (3) the basic agent plus communication skills (BC), and (4) the basic agent plus location and communication skills (BLC). The latter 3 agents have all of the capabilities of the basic agent plus additional capabilities. In describing these agents we will begin by describing the basic agent.

Agent B: Basic agent – The basic agent has a very primitive mental model of the warehouse (e.g., it has an internal symbolic representation of the warehouse which contains general information about its position in the warehouse, the location of stacks, and the location of the conveyor belt). This mental model is incomplete in that, while the agent starts with a basic model of what the warehouse looks like, it does not start out knowing the specifics of a particular warehouse. The difference is between knowing that there are stacks and items in stacks and knowing the actual contents of a specific stack. The basic agent starts by knowing the former and not the latter. The basic agent also has a generic understanding of the task (get orders, fill orders). It does not have any specific knowledge prior to taking an order of what items can be or are on the order or whether the items needed to fill an order are located in the warehouse. Two or more basic agents in the same organization differ from each other in the particular orders they acquire and in the timing with which they acquire the orders.

The basic agent has four problem spaces (knowledge domains) that relate to doing the warehouse task. These domains contain general information of the warehouse, manipulation of items in stacks, movement of the agent from one location to another, and acquisition of orders. Associated with each of these problem spaces are a set of operators which define the actions that the agent can take. The basic agent uses its mental model of the warehouse to determine which of these actions to take. The relationship between problem spaces (boxes) and operators is given in Figure 2. The first three operators in the warehouse problem space (manipulate stacks, go to location, and take orders) are "mental actions" in that they do not directly correspond to an external action, but when invoked by the agent determine which knowledge domain is needed to do that sub-task. In contrast, all other operators correspond to external actions. These operators, when instantiated, direct the agent's external actions. For example, the operator "move agent" can be instantiated as either move left or move right. If instantiated as move left, a direction to Soar-IO is issued that causes the actual agent in the actual warehouse to move left.[17] Within the agent's mental model, the symbolic agent is also moved to the left.

Figure 2: Problem spaces and operators for basic agent



The basic agent can take any of the following seven external actions:

(1) **move-to-the-left** – Agent is at a particular stack and moves to the stack immediately to the left.

(2) **move-to-the-right** – Agent is at a particular stack and moves to the stack immediately to the right.

(3) **move-item-to-left-stack** – Agent is immediately in front of a stack and moves the top item on the stack to the top of the stack immediately to the left.

(4) **move-item-to-right-stack** – Agent is immediately in front of a stack and moves the top item on the stack to the top of the stack immediately to the right.

[17]Soar-IO stands for Soar Input/Output. IO routines are those procedures which allow the program to access information from, and provide information to the outside.

(5) **move-item-to-conveyor-belt** – Agent is immediately in front of a stack and moves the top item on the stack to the conveyor belt. When this action is taken, the moved item on the order is considered to be filled.
(6) **take-order** – Agent is immediately in front of the order stack and takes the top order from the order stack. When the order is taken, the agent can then proceed to fill it.
(7) **wait** – Agent is in a queue in front of a stack and the agent waits rather than continuing to move left or right. The agent only waits if there is another agent in front of it and if it wants something in that stack.

When the agent is in front of a stack it examines the contents of the stack to determine whether the item it is looking for is in that stack. If the item is in that stack, then the agent manipulates the stack until the item sought is on top of the stack. The top item, if it is not the desired item, is always moved to the top of the stack immediately to the left or right of the current stack. If the top item is the desired item it is taken from the top of the stack and placed on the conveyor belt. When the agent goes to a location, it goes either to a stack or to the posted orders. When it reaches a location an agent can always view the contents of the stack even if there is another agent in front of it. An agent will only wait at a location if it wants something at that location (another order or an item). The basic agent solves the task (i.e., fills its orders) by systematic search. It has no item location memory and cannot send or receive communications from other agents.

In taking these actions, there are times when an agent may have options. There are several typical cases when this can happen. For example, an agent may be able to move to the left or move to the right. In another case, an agent may be able to move a blocking item from the top of the stack to the stack to the left or right. Such choices are resolved by arbitrarily setting an explicit goal for a predefined choice. Thus the basic agent will choose to move itself right rather than left and it will choose to move the blocking item to the left rather than to the right.

We define three additional agents by augmenting the basic Plural-Soar agent with additional capabilities that reflect the two primary cognitive capacities of interest in this research: location memory and communication skills. An agent either has or does not have one or both of these capabilities. An agent without either of these capabilities is the basic agent we have just described. As we have noted, a basic agent knows what is in a stack only when it is located immediately in front of it. Upon leaving the stack, a basic agent promptly forgets the contents of the stack; it has no long term memory of where things are located in the warehouse. Furthermore, the basic agent has no communication skills: it can neither ask nor answer questions, nor give advice and so forth. The basic agent acts as a simple robot, mechanically filling orders, completely independent of what it did last except for where it happens to be physically located.

Agent BL: Basic Agent plus location memory – In contrast to the basic agent, an agent with location memory retains its mental model of the contents of the stacks it encounters in the warehouse. With location memory, an agent updates its mental model of the warehouse as it leaves a stack. Thus, the BL agent remembers what is in a stack only when it departs from the location. The BL agent does not remember what items it has moved nor where it has moved those items.

Agent BC: Basic Agent plus communication skills – This agent differs from the basic agent in that it can update its mental model of the warehouse on the basis of what other agents tell it as well as on what it directly encounters. Therefore, the BC agent can learn through its own experience or through the experience of other agents capable of communicating. The BC agent has two additional operators available to it in the warehouse problem space: ask questions and answer questions. When a BC agent asks a question, it effectively broadcasts to all other agents at once: "Does anyone know the location of item *x*?" where *x* is the item on the order it is trying to fill. When the BC agent answers a question, it broadcasts to all other agents: "Item *x* is in stack *y*", where *y* is the location of the stack that the agent thinks contains item *x*. Since it has no location memory, the BC agent can communicate only if it is immediately in front of a stack containing the requested item. The BC agent does have a question memory; that is, it remembers the broadcasted questions and will respond if it ever encounters the particular item. As modeled, the BC agent is "averse" to physical effort; an agent with communication skills will ask, rather than search for items. Corresponding to the two operators are two external actions that the agent can take:

(8) **ask-question** – Agent has an unfilled item on an order and asks other agents if they know the location of the item. Asking is done via broadcast: Does anyone know the location of item *x*?

(9) **answer-question** – Agent believes it knows the location of an item and communicates that location. Answering is done via broadcast: Item x is in stack y. Agents only provide information on the specific item requested.

Agent BLC: Basic Agent plus location memory and communication skills – The BLC agent combines the capabilities of the BL and BC agents. Thus, BLC agent both remembers the contents of stacks that it has visited and can ask and answer questions. There is an interaction between location memory and communication skill: when an agent with location memory can also communicate, it updates its mental model of the warehouse and the items in the stacks on the basis of both personal experience and inter-agent communication. As modeled, the BLC agent trusts its own memory over what another agent tells it.

3.3 Coordination scheme

Plural-Soar agents in the warehouse are part of an organization that has a coordination scheme. In this study, we concentrated on two aspects of coordination: the number of agents and the communication structure. All of the organizations that we examine in this chapter are decentralized decision structures; that is, all agents are independent and direct their own action. Thus, each agent acts autonomously, makes its own decisions, and fills its own orders. Even those agents who can request information (BC and BLC) can, and do, continue to locate items and fill orders in the absence of other agents providing them with information.

Number of Agents: The number of agents in the warehouse filling orders varies from one to five. This range is sufficient to compare the results obtained by simulation to those resulting from small group experimental communication studies (e.g., Cohen, 1962; Shaw, 1981). In the one-agent scenario, the agent fills all orders sequentially. In the multi-agent scenario, each agent sequentially takes an order; however, all agents can be filling orders in parallel. No constraints are placed on the number of orders that any agent can fill; thus, the number of orders filled by each agent is opportunistically dependent on the speed with which they fill orders and the sequence in which they initially acquire the orders.

Communication Structure: We examine two communication structures: no-communication and broadcast. In the no-communication structure, no agent communicates to any other agent and all agents work independently to fill their own orders. This type of structure is analogous to teams (Arrow, and Radner, 1979; Marschak, 1955; Radner, 1987). In the broadcast structure, all agents communicate to all other agents at once. Such structures are similar to the completely connected or "comcon" networks examined by small group researchers interested in communication (Cohen 1962; Shaw, 1981), the difference being that information flows from one agent to all other agents in parallel, rather than serially to one agent at a time. In the broadcast structure, the communication is effectively a one-to-*n* communication channel. We chose these two structures because they represent, in some sense, opposite ends of a spectrum of possible communication structures and are both structures commonly used by human agents in real organizations.

3.4 Measuring performance and simulations

The type of agent in a particular coordination scheme defines the type of organization. We have identified four agents and ten possible coordination schemes. By systematically varying the types of agents across the coordination schemes, we were able to examine the relative performance contribution of each capability and its usefulness in various coordination schemes. The communication structures and the types of agents overlap; that is, Agents B and BL take part only in the no-communication structure, whereas agents BC and BLC take part only in the broadcast communication structure. Consequently, the total number of organizational types that are examined is twenty. The experimental design is shown in Figure 3. Each checkmark in the figure represents a single organization and a single simulation. We expected the different types of organizations to exhibit different performance levels. There are many factors that determine an organization's performance. In order to capture different aspects of performance we used a variety of measures focusing on *time* (to complete the task), *effort*, and *process*. Each specific measure will be described in detail as it is used.

Figure 3: Experimental design

	Broadcast				
Size	1	2	3	4	5
Agent					
(1) Basic agent [B]	no	no	no	no	no
(2) + Location memory [BL]	no	no	no	no	no
(3) + Communication Skills [BC]	yes	yes	yes	yes	yes
(4) + Location memory + Communication Skill [BLC]	yes	yes	yes	yes	yes

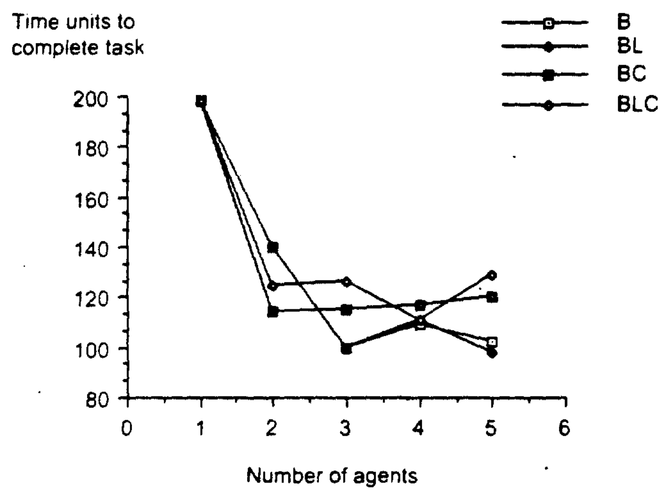
We conducted a series of 20 simulation experiments one for each of the organizational types described in Figure 3. These simulations indicate how it is possible to use this approach to determine if the capability of the agents determine which coordination scheme is most effective. In other words, simulations such as those described herein can be used to examine whether there is a tradeoff between organizational coordination and individual capability. As we have noted, each simulation experiment consisted of running a set of Soar agents that are "working together" to accomplish the warehouse task. In contrast to the Monte Carlo approach more common in organizational simulations, we simulated each organizational type only once. This is because in the Plural-Soar Model there are no dominant stochastic elements.

4. Results

Typically in organizations, for many tasks – especially tasks which admit a division of labor– there are economies of scale, in that the unit cost of production decreases as the size of the operating unit increases (Chandler and Dames 1980). In our simulations, organizations exhibit this simple, but commonly encountered aspect of organizational behavior (see Figure

4). In order to determine this, we defined each external action an agent can take as requiring one time unit for execution.

Figure 4: As more agents are involved, time decreases

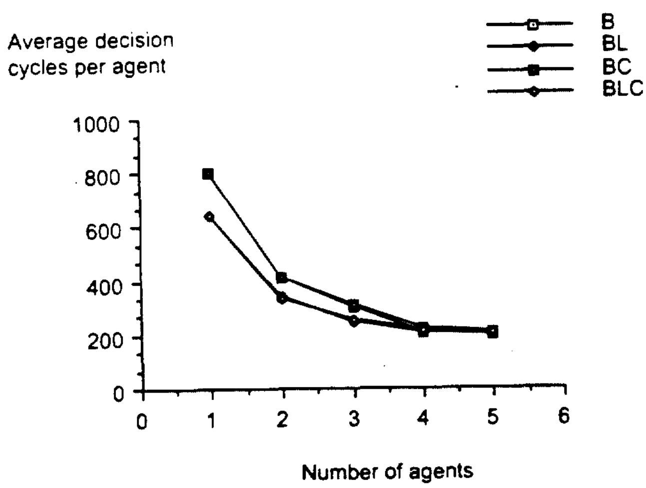


For each agent in an organization, we summed the number of time units spent doing the task. The length of time it took the organization to perform the task was defined as the maximum number of time units spent by any agent. Figure 4 shows the time it took an organization to perform a task. It can be seen that while two agents take less time to perform the task than one agent, adding a third may not save the organization additional time. Adding extra agents, however, typically decreased the average number of time units spent by any specific agent.

The general, albeit naive, expectation is that the more agents that are involved in a task, the less effort each agent has to expend, both cognitively and physically, simply because there is less for each agent to do. Of course, as Brooks (1982) points out, assigning tasks to a group does not proportionally decrease individual effort. Coordination, extra meetings, and so forth, all serve to make group work more complex. In a limited system, like Plural-Soar, many of the factors that make groups less efficient are not possible. Our agents, after all, cannot call meetings. Thus, we expect that effort average per agent will indeed decrease as more agents engage in the task, but that the decrease will not be linear. Regardless of the number of agents, there will be certain cognitive and physical start-up costs that an individual will engage in regardless of the number of agents. In addition, communication is also expected to reduce the amount of effort that agents must expend. Indeed, various communication technologies, in promoting the sharing of information, should improve organizational performance by reducing the number of times the same problem needs to be solved. Thus we expect that agents who can communicate will exhibit less cognitive and physical effort than other agents. We measured "cognitive effort" as the average number of decision cycles per agent in the organization.

Our expectation was born out: cognitive effort decreased as more agents were engaged in the task (see Figure 5). Further, agents who could communicate required fewer decision cycles to perform the task than did those agents who had no communication skills, provided there was a sufficient number of communicating agents. Even though the effect was in the right direction, communication had little total effect on overall cognitive effort.

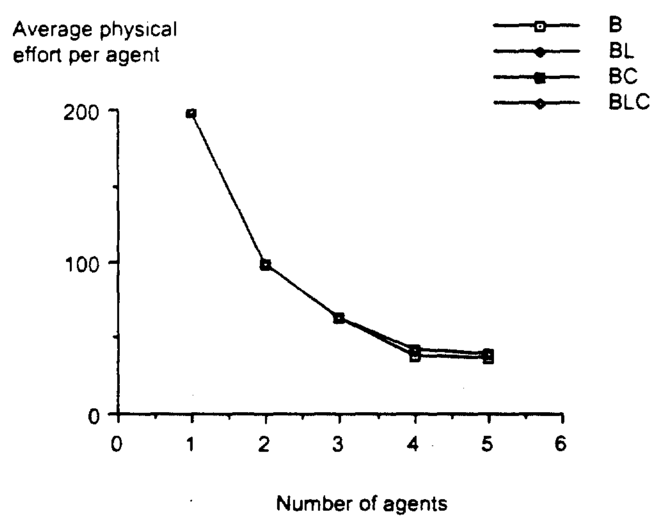
Figure 5: Cognitive effort decreases as more agents are involved.



Since we implemented communication in a very rudimentary way, this could have minimized the effect of communication. The main reason communication had little impact was that the distribution of items in orders across stacks was such that the likelihood that any agent would have encountered a needed item (and could thus tell another agent about its location) was extremely low. Consequently, although the agents could communicate, they rarely had anything of value to say.

Similar results emerge when physical effort is examined. With a fixed number of orders, the more agents, the fewer orders any particular agent will need to fill. Since agents can move items into unexpected locations, increasing the number of agents having location memory will in effect increase the likelihood that any individual agent's memory is incorrect. We defined "physical effort" as the number of moves the agent made (steps to the right and left), plus the number of times the agent moved an unwanted item to the right or left, plus the number of times the agent moved a wanted item to the conveyor belt. We found that as the number of agents in the organization increased, the average physical effort per agent decreased (see Figure 6). There is, however, hardly any difference between the various types of agents. This is due to the very low ratio of sought items to unsought items per stack. Despite the fact that many items are moved, very few of the moved items were those required by agents to fill orders.

Figure 6: Agents expend less physical effort as more agents are involved



We further expected that the idle time per agent would increase with the number of agents in the organization, due to the delay in the queues to get access to stacks. Communication skills and location memory, however, should somewhat reduce the amount of waiting as agents would be more likely to go to the stack where an item was located, rather than waiting at every stack along the way to determine if the needed item would be in that stack. We found that as the number of agents increased, more time was spent waiting (see Figure 7).

Surprisingly, we found that for three or more agents, communicating agents spent more time waiting than did non-communicating agents. This result was an unintended consequence of making the communicating agents "lazy." Recall that agents prefer to ask than to search. Thus, when an agent obtains an order, it remains in front of the order stack and broadcasts a request for information on an items location. This behavior increases the waiting time for other agents who cannot take an order until the questioning agent moves away from the order stack. No agent in this situation waited for anything other than to take an order. This is an example of an unintended social consequence where individually rational behavior results in dysfunctional social behavior.

Figure 7: Agents spend more time waiting the more agents are involved

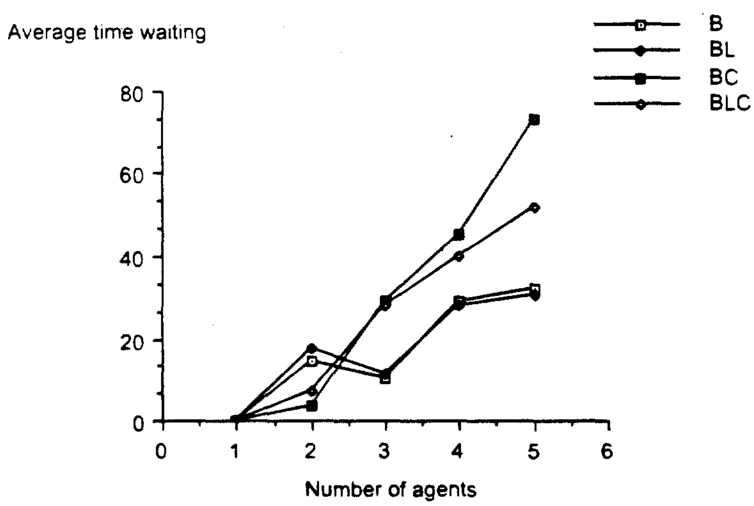
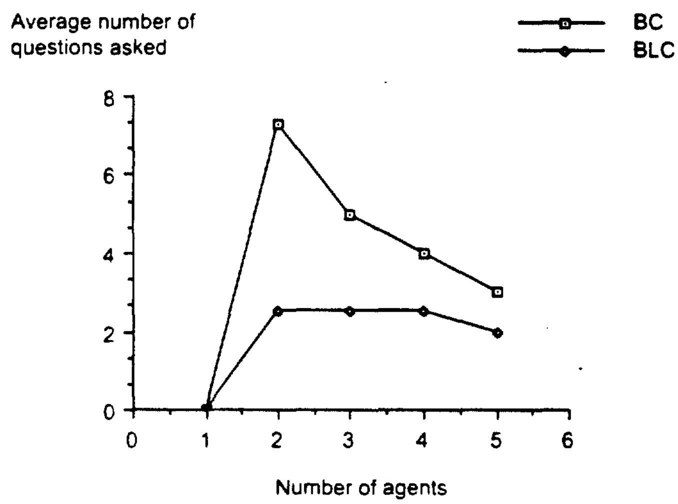


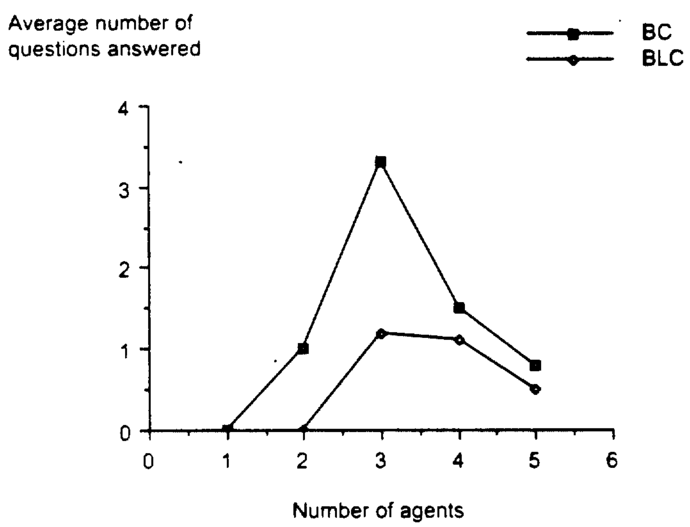
Figure 8: Agents ask more questions the more agents are involved



The average number of questions asked per agent decreased as the number of agents increased, because agents only asked questions when they obtained a new order. As the number of agents increased the number of orders per agent decreased and, so consequently did the number of questions (see Figure 8). We expected that agents with location memory would, on average, ask fewer questions than those without, due to the interaction of two preferences: the agents prefer to ask rather than to search for items (thus increasing the number of questions asked in the no-memory case) and the agents prefer to follow their memory from observation or instruction rather than ask for an item (thus reducing the number of questions in the location memory case). Both behaviors were exhibited (see Figure 8).

Finally, we expected the average number of questions answered per agent to decrease as the number of agents increased. Basically, the total number of questions asked was independent of the number of agents, so the number of questions to respond to was constant. However, as the number of agents increased, each agent would spend more time waiting and would work on fewer orders. Both of these factors contrived to decrease the amount of information available to the agent. Since agents could only answer questions when they knew where the requested object was located, the average number of answers per agent should decrease as the number of agents increased.

Figure 9: Number of questions answered depends on number of agents



In Figure 9, however, we see a curvelinear relationship between questions answered and organizational size. While the argument just provided holds for four and five agents, it does not hold for two agents.

In general, the relationships we observe are not linear. This corresponds to the fact that there are decreasing returns from scale. Moving from one agent to two agents vastly simplifies the task from both the organization's and the individual's perspective. Additional agents, however, confuse the issue. That is, as more agents are added, they begin to get in one another's way.

5. Discussion

In this chapter, we have shown how it is possible to study organizational phenomena by taking a set of generally intelligent artificial agents, placing them within an organizational structure with the constraints that it places on behavior, and providing them a task to perform collectively. Even an organization of extremely primitive agents, like those employed in this study, when engaged in a joint task nevertheless exhibit some of the basic phenomena that characterize life in an organization. Since the agents in this study are primitive, the organizational results should be viewed as indicative of the kind of phenomena that can be investigated using this approach and not modeled results about how organizations actually behave. Despite the simplicity of the simulation, the results qualitatively match observed organizational behavior in several ways; that is, the number of agents involved in the task makes a difference to both individual and organizational performance. In addition, the model provides intriguing hypotheses that can be further explored; for example, the suggestion that increasing agent capability is not necessarily advantageous to the organization. Finally, and most important, this research demonstrates the viability of the approach. We have constructed a Soar system, albeit restricted in size and function, that solves a version of the task. This suggests that we have the basic apparatus for running concurrent, interactive Soar processes that can provide data on the three-way relationship between individuals, tasks, and organizational design.

We have demonstrated that the effects of location memory and communication can be studied separately. The negligible impact of location memory in this research is due to the infrequency with which required items appeared in the stack. Thus, future studies should vary the likelihood of the required item appearing in a stack. Further, in this investigation communication has a small, but noticeable impact. This is attributable to the fact that only the most rudimentary aspects of communication were implemented in the system. While there are many ways of enhancing the communication model, we expect that it will be important to consider the differences due to one-to-one and broadcast communication.

There are a large number of ways in which the model of the agent can be extended to make it increasingly valid, such as to incorporate multiple ways of communicating, varying the types of information to communicate, and simulating information loss through oblivion. The extension of the model is therefore one of extending the artificial agent's knowledge and range of possible behaviors to coincide with visual and verbal protocols of humans performing a similar task. It is also important to conduct a series of human studies that focus on how humans behave and on how humans adapt to task variations and constraints — that is, how humans learn and acquire task-related knowledge within a group setting. We expect that observation of human subjects performing this task will refine our model.

While Plural-Soar as a model of the social agent is restricted, the warehouse task is already sufficiently complex to elicit a wide range of coordination and communication behavior. A further advantage of this task is that it can be implemented by simulation or by physical configuration without any major loss of information. Thus, human experimental studies and the simulation can be made empirically symmetrical: manipulations and alternative configurations can be made both in the human warehouse and the computer warehouse. In fact, experimental studies can be done in which some of the agents are humans and some simulations. Thus, this task facilitates important collaboration between observation and simulation with predictions and insights gained by jointly conducting behavioral task simulations and modeling those efforts.

By building the warehouse agent in Soar, we have built a social agent on top of a general cognitive agent. A large amount of research has demonstrated the validity of Soar as a viable, scientific articulation of a theory of general human cognition (Newell, 1990). Thus, by building the warehouse agent in Soar, we are importing all the complexities and realism of the Soar cognitive agent. Such a move makes it possible to incorporate all of the more established aspects of Soar, such as universal subgoaling and chunking, as well as more recent developments such as the mechanisms for recovery from error and learning declarative knowledge. Nevertheless, as Carley and Newell (1990) argued, the Soar agent is not a full social agent. A major aspect of creating a social agent is the specification of social and cultural knowledge for the agent. In Plural-Soar, we have specified only task constraints and preferences in agent behavior vis-a-vis the task. Future studies will need to provide more details to the agent on social and cultural knowledge that goes beyond the specific task.

One particular type of knowledge that will be imperative to the social agent is knowledge about the other agents in the task. That is, the agents will need a person-memory which is the agent's mental model of where other agents are in the warehouse and the correctness of the information given it by other agents. Having both location- and person-memory will make it possible to distinguish the relative impact of knowing certain information and the certainty of that information, since the agent will be able to distinguish the source of its information. An agent with only location memory cannot distinguish how it acquired the information in its model; that is, the agent will weight equally direct observation and hearsay. An agent with only person-memory, on the other hand, cannot recall information it has observed. When an agent has both location- and person-memory, it will be able to distinguish how it acquired the information.

In the current system we have not taken into account all of Soar's capabilities — most noticeably learning. While some of the agents we have modeled can learn (e.g., those with location memory can learn by augmenting their memory when they see new items), they do not learn through chunking. Given the fact that items did not repeat in orders, nor in the stacks, and since chunks are built only upon returning from subgoaling, and since the agent's subgoals are directed toward finding a specific item, potential chunks would be of little use to the agent. Consequently, while there would be across-trial learning (in the case where the agent is faced with an identical warehouse), there would not be substantial within-trial learning. In future studies, it will be important to examine the impact of true learning on behavior. It will also be important to examine the relationship between superstitious learning (such as "if it happened once, it will happen again") with the agent's model of the environment (how the warehouse is stocked). We specifically identify this type of "mislearning" because it not only occurs in organizations but is also a type of mislearning to which Soar is particularly susceptible.

Even with a fully articulated agent, it will be necessary to analyze the resulting simulation tests to determine the robustness of the results in the face of small changes in the task. Such changes would include: varying the number of types of items, the number of items in an order, the number of orders to be filled, the number of stacks in the warehouse, the number of items in a stack, and the likelihood of a required item being in a particular stack.

6. Conclusion

Plural-Soar is the first system to place Soar in an organizational or social setting. Like many AI systems, Soar has demonstrated problem-solving skills; however, like most AI systems, Soar has not demonstrated social skills. Such utilization of the Soar system provides valuable insight into what needs to be done to create an artificial system that is a model of not only a cognitive agent but a socio-cognitive agent. Earlier in this chapter (Table 1), we provided a list of the type of capabilities an intelligent agent must exhibit if it is to be a social agent within an organization. Now we can re-examine these capabilities to see how much progress has been made.

As can be seen in Table 2, there are two major gaps: the Plural-Soar agents are not capable of analyzing either the task or the social environment. Associated with this fact is the point that these agents do not have a memory of task, a memory of where other agents are located, or a memory of what these other agents are doing (other than a memory that somebody asked a question). Further, the communication skills exhibited by these agents are extremely minimal and do not make it possible for one agent to supervise another agent, as it happens in organizations. We expect that major progress in using models of task-oriented cognitive agents as the basis for micro-level theories of organizational behavior will require the expansion of a system like Plural-Soar in such a way that the agents have a model of the task, other agents, and the organization.

Table 2: Exhibited social agents capabilities

Perception and Action	
Perceives the environment	x
Physically manipulates objects	x
Moves self to different locations	x
Memory	
Location	x
People	
Task	
Instruction	x
Task Analysis	
Decomposes task	
Coordinates subtasks for self to do	
Communication Skills	
Asks questions/Provides answers	x
Gives commands/Receives commands	
Talks to a single individual/Talks to a group	x
Social Analysis	
Models of other agents	
Model of organization	

From an organizational perspective, we expect that this approach will enable the development of a more cognitively motivated theory of micro-organizational behavior. This theory would reach beyond more classical information-processing models of organizations, which tend to leave concepts such as knowledge ill-defined with respect to cognition, by extending its analysis to include agent capabilities expressed directly as knowledge. A final advantage to this approach is that it focuses attention on the role of individual preferences in determining social outcomes within the organization. Consequently, this approach makes possible a theory of organizational behavior that takes into account not only agent capabilities but the preferences of the agents which determine their application of those capabilities. Since such preferences are often the result of existing norms or the accidents of history, the approach we have proposed in this chapter makes possible a truly cognitive and social account of organizational behavior.