Learning in Agent-Based Manufacturing Systems

Weiming Shen, Francisco Maturana and Douglas H. Norrie

Division of Manufacturing Engineering, The University of Calgary
2500 University Dr. NW, Calgary, Alberta, Canada T2M 1N4
E-mail: [wshen | norrie]@enme.ucalgary.ca

Abstract

Agent-based technology has been taken as an important approach for developing advanced manufacturing systems. Learning is one of the key techniques for implementing such systems. This paper proposes some learning issues in agent-based manufacturing systems for further discussion in the working group. As examples, the paper describes two implemented learning mechanisms and presents some experimental results in an agent-based manufacturing system.

1 Introduction

Recently, agent-based technology has been taken as a promising approach for developing advanced manufacturing systems (Cutkosky et al 1996; Maturana and Norrie 1996; Park, Tenenbaum, and Dove 1993; Parunak, Baker, and Clark 1997; Shen, Xue, and Norrie 1998). Such an approach provides rapid responsive and dynamic reconfigurable structures to facilitate flexible and efficient use of manufacturing resources in a rapidly changing environment.

For most application tasks, it is extremely difficult or even impossible to correctly determine the behavioral repertoire and concrete activities of a multi-agent system a priori, that is, at the time of its design and prior to its use. This would require, for instance, that it is known a priori which environmental requirements will emerge in the future, which agents will be available at the time of emergence, and how the available agents will have to interact in response to these requirements. Such problems resulting from the complexity of multi-agent systems can be avoided or at least reduced by endowing the agents with the ability to learn, that is, with the ability to improve the future performance of the total system, or a part of the system.

Many researchers have worked on multi-agent learning in different applications, such as prisoners dilemmas, predator/prey, code warriors, virtual robots, self-playing game-learners, traffic controls, with varying success, but few applications can be found in manufacturing systems. This paper proposes the learning issues in agent-based manufacturing systems for further discussion in the working group, describes our approach for enhancing the performance of an agent-based manufacturing system through 'learning from the history' based on distributed case-based learning and reasoning and 'learning from the future' through system forecasting simulation. The rest of the paper is organized as follows: Section 2 discusses the learning issues in agent-based manufacturing systems; Section 3 describes the learning mechanisms developed for the MetaMorph system; Section 4 presents some experimental results; Section 5 gives some conclusions and discussions.

2 Learning in Agent-Based Manufacturing Systems

2.1 Why Learn

Manufacturing environments are real-time, dynamic systems. There are always different problems like failures of machines, tools or transport vehicles, and lacks of required materials. On the other hand, new tools, machines or transport vehicles may be added into the working manufacturing environment. Thus, an agent-based manufacturing system must be able to adapt changing environments and handle emergent contexts. Extensive search processes are required to identify the multi-agent interactions for each problem case. These interactions give rise to observable patterns of behavior evolving concurrently within each level in the organization for different types of requirements, constraints, and agents.

According to Goldman and Rosenschein (1996), learning in a multi-agent environment can help agents improve their performance. Agents, in meeting with others, can learn about the partner's knowledge and strategic behaviors. Agents that operate in dynamic environments could react to unexpected events by generalizing what they have learned during a training stage. In cooperative problem solving systems, cooperative behavior can be made more efficient when agents adapt to information about the environment and about their partners. Agents that learn from each other can sometimes avoid repeatedly coordinating their actions from scratch for similar problems. They will sometimes be able to avoid communication at run-time by using learned coordination concepts, which is especially useful whenever they do not have enough time to negotiate.

2.2 When to Learn

In agent-based manufacturing systems, agents should learn in the following situations:

- When the system configuration changes. For example, new manufacturing resources (machines, tools, transport vehicles, workers, etc) are added to the working manufacturing system; some resources in service breakdown or are removed from the system; some other hardware or software are changes or updated. In any case mentioned above, each agent should learn about the changes of the working system and update its knowledge about its environment and other agents.
- When some failures occur in the system. Failures can occur either during the design process or by testing/simulating/manufacturing the designed artifact. Both situations represent learning opportunities.
- When there exist differences on a same design or manufacturing object. Differences, especially between different points of view, are an important source of conflicts. Learning triggered by noticing differences helps anticipate and alleviate conflicts.
- When a project or a task is terminated with success. In manufacturing systems, a successful process planning (an efficient combination of the manufacturing resources for a special manufacturing task) is a good learning case.
- When there is a need to improve abilities. The need to improve designs, design processes or manufacturing process plans can be translated into a requirement to improve agent abilities. This means that an agent uses the learning about design situations, manufacturing resource availability or other design/resource agents for more informed decisions in the future. Observing patterns, dependencies or other relations can be useful even though it is not motivated by events falling in any of the previous four categories (situations).

2.3 Where to Learn

In agent-based manufacturing systems, the way that agents make information available to the other agents determines where the learning opportunities arise. Varying the representation and the context in which the knowledge is communicated favors some of the learning types in comparison to the others.

Direct communication implies transfer of knowledge between agents through direct channels. The knowledge is encapsulated in a message with a special format and sent to other agents. The transferred knowledge is not altered in this process. When an agent receives this type of message, it extracts the useful information from the message and updates its related knowledge bases. This type of communication can be found in the DIDE project (Shen and Barthès 1997).

Indirect communication assumes the presence of an intermediary agent, conveying information from one agent to another. Knowledge may be pruned or abstracted in this process and may also be submitted to ontological transformations. Examples of this type of intermediary agent are the facilitator in PACT (Cutkosky et al 1993), the mediator in MetaMorph (Maturana and Norrie 1996) and the design board in SiFAs (Grecu and Brown 1996).

Some multi-agent learning systems are implemented with training agents (tutor agents). In this case, the training agents or tutor agents are the main knowledge source of learning for the system.

2.4 What is to be Learned

What kinds of knowledge can, and should be learned in agent-based manufacturing systems? The most important include the successes and failures of the past, the usefulness of different pieces of knowledge with respect to different tasks and situations, the capabilities and accountability of other agents in the system, and the relationship between the multi-agent system and its environment.

While the range of the learnable is fairly wide, there are areas where learning might have a serious impact for manufacturing, and for agent interactions. Some of the following types of learning target the level of manufacturing resource allocation, while others target the meta-level of agent interaction:

- *Combinations of manufacturing resources for specific tasks:* These combinations of resources at the group level or the system level can be learned as cases for manufacturing process planning, which may be used in the following process planning and scheduling so as to reduce the communications and negotiations among resource agents.
- *Manufacturing system's behavior:* By learning the system's behavior and propagating it to the near future through some simulation forecasting mechanisms, the system is allowed to learn from future emergent behaviors so as to prevent the unforeseen perturbations and changes in production priorities on the shop floor.
- *Support in favor of or against a decision:* Rationale can primarily help in weighing proposals from different agents and in deciding which agent should revise its decision.
- *Rules:* As agents have different functionality and points of view, sharing of rules is desirable among agents having the same target or at least the same domain.
- *Preferences:* If agent preferences are known, they help set up better initial proposals made by one agent to another.
- *Preconditions and postconditions for rules, actions and tasks:* This type of learning is important for a better adaptation of decisions to specific contexts. Agents decide by themselves when to act, and therefore should be able to recognize favorable situations to get involved in the design.

- *Prediction of decisions of other agents:* This amounts to building a behavioral model of other agents, by relating factors which influence an agent with the agent's decisions.
- *Types of conflicts:* Conflicts can be seen, indexed and classified differently by the different types of agents. Enabling agents to classify and recognize conflicts enhances the power of the conflict anticipation and resolution mechanisms.
- *Heuristics to solve conflicts and to negotiate:* These heuristics, being less dependent on the specific functionality of an agent, can be learned or transferred from one agent to another and refined according to local necessities.

2.5 Some Related Projects

Learning in multi-agent systems is becoming increasingly recognized as a very important topic. Some researchers have applied Machine Learning techniques for the multi-agent learning systems, while others have proposed new approaches especially for multi-agent learning (Tan 1993; Sandholm and Crites 1996; Byrne and Edwards 1996; Nagendra Prasad, Lesser, and Lander 1997; Ohko, Hiraki, and Anzai 1996). However, only a few examples can be found for agent-based manufacturing systems. Here we review only two interesting projects related to the learning in agent-based manufacturing systems.

Much of the literature of multi-agent learning relies on reinforcement learning. Gu and Maddox (1996) developed a learning model called DRLM (Distributed Reinforcement Learning Model) that allows distributed agents to learn multiple interrelated tasks in a real-time environment. DRLM was implemented in a Flexible Manufacturing System (FMS) where sensors (modeled as agents) have to learn to communicate with humans about the material handling activities using graphical actions such as displays and animation.

Shaw and Whinston (1989) proposed a classifier system based multi-agent learning. Each agent contains a set of condition-action rules and a message list. The rules that can be triggered in the present cycle in an agent can bid, based on their strengths. The rule with the highest bid is chosen and executed to output a message as an action, perhaps to other agents. Its strength is reduced by the amount it bid and is redistributed to the rules that were responsible for the messages that triggered it. These rules could be distributed among many agents. An agent choosing to perform an action on the environment receives an external reinforcement. This classifier system was tested on a Flexible Manufacturing domain. It functioned in a contract-net framework where inter-agent bidding is used as a feedback mechanism that drives inter-agent learning. An agent announces a task with a series of operations to be performed on it. Other agents in the system bid for performing this task based on their past usefulness in performing similar jobs, the capability for the announced job and their readiness for the job. The winning agent increases its strength by an amount proportional to the bid and the agent announcing the task decreases its strength by the same amount. This represents a kind of market mechanism where the announcing agent pays the winning agents for its services by the amount bid. In addition, the agents themselves can use genetic operators to improve and reorganize their own capabilities for performing various jobs. This type of learning is purely local and does not involve any cooperative control component. In the Flexible Manufacturing domain, the genetic learning capability can be mapped on to the need for reconfiguration to avoid unbalanced utilization of high-performance cells.

3 Learning in MetaMorph

3.1 MetaMorph Project

MetaMorph is an agent-based architecture for intelligent manufacturing developed at The University of Calgary using the mediator-centric federation architecture (Wiederhold 1992). The architecture has been named MetaMorphic, since a primary characteristic is its changing form, structure, and activity as it dynamically adapts to emerging tasks and changing environment. In MetaMorph (Maturana and Norrie 1996; Maturana 1997), Mediator is a distributed decision-making support system for coordinating the activities of a multi-agent system. This coordination involves three main phases: (1) subtasking; (2) creation of virtual communities of agents (coordination clusters); and (3) execution of the processes imposed by the tasks. These phases are developed within the coordination clusters by distributed mediators together with other agents representing the physical devices. The coordination clusters are initialized through mediators, which can dynamically find and incorporate those other agents that can contribute to the task. The MetaMorph architecture has been used for implementing a distributed concurrent design and manufacturing system in simulated form. This system is composed of the following modules: an Enterprise Mediator, a Design System, two Shop Floor modules, and a module for execution control and forecasting. The design system module is developed for retrieving design information and requesting manufacturability evaluations through the enterprise mediator. A detailed description of the MetaMorph project can be found in (Maturana and Norrie 1996).

For a mediator to learn about its environment, other mediators and agents, a simple registration mechanism is used to capture, manager and update its knowledge which is then reused in its future interactions and coordinations with other mediators and agents. This paper is focused on two learning mechanisms implemented in MetaMorph multi-agent manufacturing system to enhance the system's performance and responsiveness. First, a mechanism that allows mediators to learn from history is placed "on top" of every multi-agent resource to capture significant multi-agent interactions and behaviors. Second, a mechanism

for propagating the system's behaviors into the future is implemented to help mediators "learn from the future." The combined action of these two learning mechanisms provides a promising approach for multi-agent manufacturing learning.

3.2 Learning from the History

The context of manufacturing requests is established under static and dynamic variations. Static variations relate to the physical configuration of products. Dynamic variations depend on time, system loads, system metrics, costs, customer desires, etc. These two main sources of information relate to a wide spectrum of emergent behaviors, which can be separated into specific behavioral patterns. A 'learning from the history' mechanism based on distributed case-based learning approach was developed during the MetaMorph project for capturing such behavioral patterns at the resource mediator level and storing in its knowledge base. Such knowledge is then reused for afterwards manufacturing requests by an extended case-based reasoning mechanism.

A manufacturability or manufacturing request sent to a resource community is first filtered by the respective resource mediator to decide whether the request can be recognized as associated with a previously considered product or is unknown. If it is recognized, the resource mediator retrieves the learned patterns to send to a selected group of agents identified in the patterns. For an unknown request, the resource community's mediator uses its standard matchmaking actions to specify the primary set of resource agents to be contacted regarding their capability to satisfy this request. The solution to this unknown request then proceeds through the propagation of coordination clusters and decision strategies previously defined. During this process, the resource mediator involved learns from partial emergent interactions at the coordination cluster level. This learning is distributed among several coordination clusters. The plan aggregation process then enables the classification of various feature-machine-tool patterns, which are encoded and provided to the community's mediator for storage and future reuse.

The approach described above was implemented to show proof-of-concept of distributed deliberative learning for advanced manufacturing systems. However, the mechanisms are centralized through the dynamic mediators and will become increasingly slow for larger amounts of information, i.e. with increased product types and larger factory floors. The mechanisms in this learning system would, in such cases, need to be replaced by other mechanisms which allow for storing, pattern matching, and rapid information retrieval in a single action.

3.3 Learning from the Future through Forecasting

In MetaMorph, the system's behavior is extended into the near future for learning from future emergent behaviors through the implementation of simulation forecasting mechanisms.

In conventional systems, the effect of unknown future perturbations can only be estimated through probabilistic mechanisms that add in some perturbations to the scheduling calculations. MetaMorph used a 'learning from the future' mechanism to uncover dynamic interactions likely to perturb plans and schedules during execution stages. This system maintains a virtual model of the shop floor, which can be used as the base for propagating the system's behavior into the near future. The system also maintains performance evaluation parameters to accept or reject future interactions.

Shop floor mediators monitor system status by conducting global performance evaluations. Various forecast-triggering and adjusting parameters are used based on system load, extension of the scheduling horizon, and adjustment periodicity. The thresholds for triggering are arbitrarily set by the users according to high-level policies.

A composite function called System Heuristic Function is used to partially measure the system's performance. The function measures average flow time, maximum completion time, average due time, number of late jobs and total number of jobs, as shown in the following equation.

$$H = w_f * \bar{T}_{flow} / \bar{T}_{due} + w_c * T_{maxcpl} / \bar{T}_{due} + w_j * J_l / J_t$$

where:

w_f – average flow time weight

w_c – maximum completion time weight

w_j – late jobs weight

$\bar{T}_{flow}$ – average flow time

$\bar{T}_{due}$ – average due time

T_{maxcpl} – maximum completion time

J_l – number of late jobs

J_t – total number of jobs

The function value H is called the system heuristic. A lower H value indicates that the system has a higher performance with a lower cost. Its primary value is for assessing deviation towards an undesirable situation of increasing production time and lateness. Each component in this function is weighted according to the system's management policies. Other components can be added to this function so as to measure system's performance more completely. The heuristic ratio (R_h) is calculated as follows:

$$R_h = \frac{H'}{H}$$

H'–forecast simulation performance

H – system performance before adjustment

Lower and upper thresholds ($-\sigma$, $+\sigma$) have been

used to define acceptance and rejection heuristic-ratio regions. These thresholds define a square-shaped region with an amplitude equal to 1 (dimensionless). For ratios falling within the square region, the system's current state of commitments is accepted and remains unaltered. If the ratio falls outside the square region, the forecast behavior (scheduling and activities) is accepted to replace the current scheduling state.

Forecasting the system's behavior consists of capturing the current state of the multi-agent system (plans) and simulating it for a period of time into the future. The forecasting process is then a simulation of the future interactions of the intelligent agents' commitments.

The forecasting simulation period is established by the system's manager according to high- level policies. However, the determination of this parameter may be customized to automatically adapt to the system's requirements. There are two approaches that can be used for the forecasting period:

(1) The forecasting simulation may be run for a period of time sufficiently large to permit the completion of every committed job. But for large simulation periods, this criterion incurs accumulative deviations, since jobs may arrive in the system during the forecast period that affect intermediate allocation slots. Because these intermediate allocation slots are not preempted during simulation, this approach only produces rough estimations.

(2) The second approach attempts to avoid the accumulation of deviate behaviors, limiting the forecasting simulations to short periods of time only. This approach is more accurate, since the duration of the simulation is much less than the jobs' arrival frequency. Here, the multi-agent system can easily be adjusted while including the intermediate requirements.

Whichever the approach, the forecast produces an estimated heuristic value, which is used to decide whether to introduce the forecast schedule tuning or to continue with the existing system's scheduled commitments.

A detailed description of above mentioned learning mechanisms in MetaMorph can be found in (Maturana 1997).

4 Experimental Results

4.1 Learning from the Future

Several test cases have been used for verifying the efficiency of the learning mechanisms implemented. Here is a typical case with multiple shop floors and multiple products. The simulated manufacturing system consists of two shop floor resource communities. Each community has three primary machine resources, namely, Vertical Machining Centers, two secondary machining resources, namely, Internal Grinders, and fifty tooling resources to be used with those machines. Each resource is represented as a resource agent with appropriate knowledge and reasoning mechanisms. The primary resources constituted highest level of production hierarchy, while the secondary ones represented lowest level. Each community is coordinated by two types of dynamic mediators, one for machining resources and other for tooling resources. The dynamic mediators can recursively create new coordination clusters, clone agents and themselves to provide the parallelism required by a planning task(s). The dynamic mediators in turn are coordinated by one static matchmaker mediator which assumed the role of making multi-objective decisions.

A product mix comprising 100 of each of three part types, namely, Bearing Cover, Electro-Magnetic Insulation (EMI) Housing, and Guide, were to be manufactured in two shop-floor areas using an AGV transportation system to handle raw materials and semi-finished products.

Shop floor 1 included 3 vertical machining centers, 2 internal grinders, 1 surface grinder and 43 tools. Shop floor 2 included 3 vertical machining centers and 35 tools. A profit margin of 35% of normal production cost was specified.

Once the production orders were placed in the design system interface, the static mediator made repeated broadcast requests to each shop floor involved. Since each offered a different cost, the static mediator applied a final integration of plans to determine the best one to manufacture the products. The final scheduling configuration established that 190 parts were produced at shop floor 1 and 110 parts at shop floor 2. Shop floor 2 only manufactured EMI housing and Guide products. The Bearing cover product was only produced at shop floor 1.

Table 1 shows combined heuristic metrics of Average Flow Time (AFT), Maximum Completion Time (MCT), and Transportation Cost (TC) for both shop floors. There were 8 jobs in all that were tardy, showing that production capacity for these parts had been more than fully loaded.

AFT (min.)	MCT (min.)	TC ($) (Max.)	(Min.)
414.06	7164.81	50.21	25.39

Table 1: Experiment Results

During this production, the forecasting simulation proved to be a robust system for projecting the behavior and allowing adjustments to be made to schedules when needed, so as to keep the system from deviating into undesirable schedule mis-match regions. The forecasting was triggered 8 times through the entire planning process, as shown in Figure 1.

Figure 1 shows the heuristic (performance) trends for the promissory and forecasting models. A plan-refining pattern can be observed from this figure. Both promissory and forecasting heuristics (performances) had a natural tendency to diverge from each other. However, the plan-refining procedure (i.e., the adoption of the forecasting schedule) tended to counteract such divergence. Since

plan refinement depends on the acceptance threshold. plans were not always refined after each forecasting simulation. Nevertheless, the promissory plan level was progressively adjusted with the forecasting performance in such a manner as to maintain the promissory planning within realistic levels.

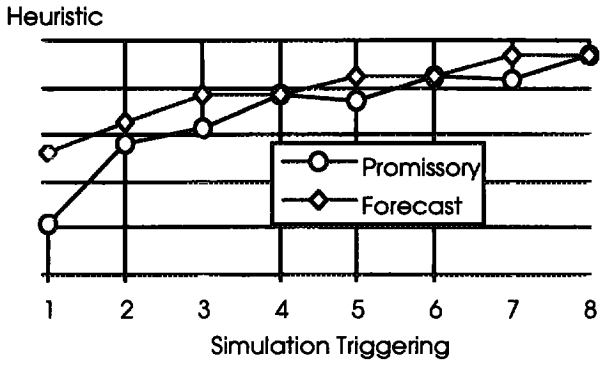

Figure 1: Triggering Thresholds

Figure 1 also shows that the forecasting heuristic is always higher than the promissory heuristic (i.e., the forecasting cost is higher than the promissory cost), because the promissory heuristic reflects the ideal system performance while the forecasting heuristic reflects the near realistic system performance by taking into account the shop floor interactions, actual production and transportation time. Dynamic interactions among transportation agents (AGVs) on the shop floor were the main cause of forecasting changes (delays) on the schedule. Three situations including shop floor traffic, collision avoidance and batch size accumulation were the primary sources of transportation delays.

4.2 Learning from the History

At the beginning of the experiments, the community's mediator does not have any organizational knowledge of relationships among resources or between resources and manufacturing requests. Agents work in a blind-search space and used coordination cluster propagation mechanisms to dynamically establish efficient relationships. Each community mediator is progressively learning emergent patterns among the shop-floor resources (machine and tools in particular). These learned patterns are then used to experiment on the behavior of the system when the community's mediators are modified to act in learning mode (knowledge reuse mode).

Each product's feature is progressively associated with a set of emergent relationships. Within each set, individual relationships offered a specific dynamic production cost. Resource agents within each relationship reevaluated individual manufacturing costs to build subsequent manufacturability evaluations. Each manufacturing cost was negotiated among agents within coordination clusters.

Each such relationship is associated with a feature related to a set of manufacturing parameters. For this experiment, direct encoding for each relationship is used, but in future implementations of this learning model, new types of learning systems (based on neural network, fuzzy logic, or tables) could be used to learn and store emergent relationships. Any such learning system must be continuously training and adjusting its knowledge.

In the experiments, the learning ability of the community's mediator is always active to enable it to capture emergent relationships. Learning mechanisms are embedded in the qualitative and quantitative evaluations. To set the system to reuse learned patterns, the reuse learning-mode button (on the shop floor's graphical interface) should be activated by the user to switch the system from the normal mode to the pattern reuse mode.

The multiple shop floor and multiple product experiment case as described in the previous subsection was used to test the performance of the system under the knowledge reuse conditions. First, shop floor 1 was swapped to the learning mode while shop floor 2 was maintained at the normal reasoning mode. This setup did not show any improvement in the task resolution speed of the system, since each integration and evaluation of final costs at the static mediator level was halted until shop floor 2 was ready to answer its final plans and bids. Since both shop floors take the same amount of time to solve tasks, there was no apparent improvement in the overall system performance.

A second setup was experimented with for both shop floors operating in learning mode. In this experiment there was an amazing improvement in the speed of resolution of the system. In learning mode, the system creates task solution 100 to 500 times faster than the normal task-resolution mode.

5 Conclusion

Learning is one of the key techniques for implementing efficient agent-based manufacturing systems. This paper proposes some issues related to the learning in agent-based manufacturing systems for further discussions in the working group.

A learning mechanism for identifying agent-based manufacturing system organizational knowledge and selective interaction propagation from emergent system behavior has been proposed. This mechanism enhances coordination capabilities by minimizing communication and processing overhead, facilitates distributed, parallel depth-first search, and therefore enhances the performance of the agent-based manufacturing system. Though this learning model has been implemented in a distributed mediator architecture that is part of a concurrent design and manufacturing system, it is generic and can be applied to other areas as well.

A mechanism for learning from the future through forecasting has also been developed for dynamically adjusting distributed schedules and planning in a multi-agent manufacturing system. Experimental results show the value of this approach for adjusting and enhancing the

performance of the agent-based manufacturing system.

Learning in agent-based manufacturing systems is an important, but difficult problem because of the complexity of dynamic manufacturing environments. Both theoretical and experimental research works are to be done in this area. The learning mechanisms proposed and developed in this paper are simple but effective approaches for enhancing the performance of agent-based manufacturing systems.

References

Byrne, C. and Edwards, P. 1996. Refinement in Agent Groups. In Weiss G. and Sen, S. (Eds.), *Adaption and Learning in Multi-Agent Systems*, Lecture Notes in Artificial Intelligence 1042, pp. 22-39, Springer-Verlag.

Cutkosky, M.R., Engelmore, R.S., Fikes, R.E., Genesereth, M.R., Gruber, T.R., Mark, W.S., Tenenbaum, J.M. and Weber, J.C. 1993. PACT: An Experiment in Integrating Concurrent Engineering Systems. *IEEE Computer*, 26(1):28-37.

Cutkosky, M.R., Tenenbaum, J.M. and Glicksman J. 1996. Madefast: Collaborative Engineering over the Internet. *Communication of the ACM*, 39(9):78-87.

Goldman, C.V. and Rosenschein, J.S. 1996 Mutually Supervised Learning in Multiagent Systems. In Weiss G. and Sen, S. (Eds.), *Adaption and Learning in Multi-Agent Systems*, Lecture Notes in Artificial Intelligence 1042, pp. 85-96, Springer-Verlag.

Grecu, D. and Brown, D. 1996. Learning by Single Function Agents during Spring Design. In Gero, J.S. and Sudweeks, F. (Eds), *Artificial Intelligence in Design '96*, Kluwer Academic Publishers, Netherlands.

Gu, P. and Maddox, B. 1996. A Framework for Distributed Refinement Learning. In Weiss G. and Sen, S. (Eds.), *Adaption and Learning in Multi-Agent Systems*, Lecture Notes in Artificial Intelligence 1042, pp. 97-112, Springer-Verlag.

Maturana, F. and Norrie, D. 1996. Multi-Agent Mediator Architecture for Distributed manufacturing. *Journal of Intelligent Manufacturing*, 7:257-270.

Maturana F.P., 1997. MetaMorph: An Adaptive Multi-Agent Architecture for Advanced Manufacturing Systems. Ph D thesis, The University of Calgary.

Nagendra Prasad, M.V., Lesser, V.R., and Lander, S.E. 1997. Learning Organizational Roles for Negotiated Search in a Multi-agent System. *Special issue on Evolution and Learning in Multiagent Systems of the International Journal of Human-Computer Studies (IJHCS)*.

Ohko, T., Hiraki, K., and Anzai, Y. 1996. Learning to Reduce Communication Cost on Task Negotiation. In Weiss G. and Sen, S. (Eds.), *Adaption and Learning in Multi-Agent Systems*, Lecture Notes in Artificial Intelligence 1042, pp. 177-190, Springer-Verlag.

Park, H., Tenenbaum, J. and Dove, R. 1993. Agile Infrastructure for Manufacturing Systems (AIMS): A Vision for Transforming the US Manufacturing Base. Defense Manufacturing Conference.

Parunak, H.V.D., Baker, A.D. and Clark, S.J. 1997. The AARIA Agent Architecture: An Example of Requirements-Driven Agent-Based System Design. In Proceedings of the First International Conference on Autonomous Agents, Marina del Rey, CA.

Sandholm, T., and Crites, R. 1996. On Multi-Agent Q-Learning in a Semi-Competitive Domain. In Weiss G. and Sen, S. (Eds.), *Adaption and Learning in Multi-Agent Systems*, Lecture Notes in Artificial Intelligence 1042, pp. 191-205, Springer-Verlag.

Shaw, M. J. and Whinston, A. B. 1989. Learning and adaptation in DAI systems. In Gasser, L. and Huhns, M., (Eds.), *Distributed Artificial Intelligence*, volume 2, pages 413-429. Pittman Publishing/Morgan Kauffmann Publishers.

Shen W. and Barthès J.P. 1997. An Experimental Environment for Exchanging Engineering Design Knowledge by Cognitive Agents. In Mantyla M., Finger S. and Tomiyama, T., (Eds.), *Knowledge Intensive CAD-2*, Chapman and Hall, pp. 19-38.

Shen, W., Xue, D., and Norrie, D.H. 1998. An Agent-Based Manufacturing Enterprise Infrastructure for Distributed Integrated Intelligent Manufacturing Systems. In Proceedings of PAAM'98, London, UK.

Tan, M. 1993. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In Proceedings of the Tenth International Conference on Machine Learning, 330-337.

Wiederhold, G. 1992. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38-49.