INTELLIGENT SHOP FLOOR SCHEDULING METHOD CONSIDERING REAL-TIME STATUS, DISTURBANCES, AND PERFORMANCES

Jayendran Venkateswaran, Xiaobing Zhao and Young-Jun Son*

Systems and Industrial Engineering
The University of Arizona
1127 E. North Campus Drive
Tucson, AZ 85721-0020
*Corresponding author's e-mail: son@sie.arizona.edu

Abstract: Due to the stochastic and dynamic nature of a shop floor, which cannot be perfectly predicted, the actual events in the shop can be considerably different from those specified in the original schedule. Therefore, corrective actions should be taken to enhance the degraded performance of the original schedule. This paper presents a distributed adaptive control system composed of intelligent agents that will perform adaptive scheduling and schedule execution. The major features of the proposed system include 1) the distribution of the tasks among the independent agents, 2) the development of a passive supervisory component, which gains knowledge from the system proceedings, and 3) the use of artificial intelligence techniques to dynamically map the global performance and the state of the system to the local decision making rules of the distributed agents. In this paper, the framework and design issues for the proposed system are discussed.

1. INTRODUCTION

In recent years, there has been a general trend towards the development of a good quality schedule that can be directly used to control the shop. The quality of a schedule is measured by the extent to which it satisfies its objectives (normally performances), when executed in the shop floor. Considerable research has been carried out since the early 1990's to come up with good quality schedules that can handle unplanned events that may occur in the shop. Research work has been carried out in the generation of predictive schedule to handle disturbances that occur with some level of certainty (Mehta and Uzsoy, 1998). The practice of rescheduling to handle disturbances has been proposed in which a new schedule is generated when the current schedule becomes invalid. The new schedule is generated either periodically after a set interval of time (Sabunchuoglu and Bayiz, 2000), or based on the occurrence of a disturbance or based on the performance of the shop. Adaptive scheduling technique has been proposed in which the schedule is tailored to the current state of the system. The work in the area of adaptive scheduling is classified into, those that use static rule(s) to associate the different system parameters to the scheduling policy (Adacher et al., 2000) and those in which the rules evolve (learn) from past decisions (Park et al., 1997; Maione and Naso, 2001).

The manufacturing system can be viewed as a control system that monitors, coordinates, and executes the various shop resources to achieve the system objectives. Diverse control frameworks (architectures) have been proposed over the years (e.g., centralized, hierarchical, heterarchical, and hybrid), which have a direct impact on the choice and performance of the scheduling method. Often times the control is presented in conjunction with the scheduling policies and methods, especially when the execution of the schedule is to be monitored and/or the schedule is to be modified based on the current status of the shop.

Under the hierarchical frameworks, a multi-level hierarchy of controllers exists, where controllers in different levels form a master/slave relationship. The global perspective of a higher level controller helps in the construction of a state-dependent schedule. However, hierarchical controls suffer from drawbacks such as excess computation time (in the case of frequent disturbances) and obsolesce of the status information used for schedule generation. The need for a robust system has led to the development of heterarchical control frameworks where distributed autonomous entities (control components in the same level) communicate with each other to pursue system goals (Dilts et al., 1991). The entities negotiate with each other to make local and independent decisions (Solberg and Lin 1992). Though this provides increased adaptability to changes in the shop, it becomes difficult to predict the system state or arrive at a global optimum. In addition, it is difficult to implement learning schemes since a global perspective of local decisions is absent.

Hybrid frameworks have been developed to overcome some of the drawbacks of pure hierarchical and heterarchical frameworks by exploiting the features of those frameworks. Hybrid frameworks include lower level distributed components that negotiate with each other to make decisions. They also include a supervisory controller to obtain a global perspective of the system. There exists a loose coordination between the supervisory and lower level components. Several frameworks have been presented (Adacher et al., 2000; Maione and Naso, 2001), including holonic

manufacturing systems, bionic manufacturing systems, fractal manufacturing systems. These systems better achieve robustness in the face of disturbances and strive to satisfy system-wide objectives. Although a global perspective is available for the higher level controller, a suitable scheme to relate local decisions of the lower level components with the global outcome of the system is missing. The lack of proper interrelationships may result in ineffective learning from past decisions.

Agent technology has been popularly used to implement the distributed control entities. The general issues of concern include the communication structure, the type of interaction, and the selection of agents (entities). The contract net protocol, for communication based on the negotiation procedures, enables the entities to announce a task, collect the bids from candidates and make selection among them using their decision making abilities. Several agent-based manufacturing control architectures, for pure heterarchical as well as hybrid control, have been proposed (Solberg and Lin, 1992; Adacher at al., 2000; Cavalieri et al., 2000; Maione and Naso, 2001; Odrey and Mejia, 2003). Most of these architectures identify two agents, *machine agent* (or workstation agent or resource agent) and *part agent*. Part-initiated negotiation procedures (Solberg and Lin, 1992), machine-initiated negotiation procedures (Macchiaroli and Riemma, 2002) have been proposed based on a market-like bidding scenarios. Multi-agent system by Adacher et al. (2000) identified machine agents, part agents and storage agents. They simulated 5 different negotiation procedures, with different settings of batch size, buffer size and arrival intervals to evaluate the trade off between the degree of autonomy and system performance. Maione and Naso (2001) constructed a control structure for an FMS consisting of part agents and machine agents. The decisions of the agents are based on multi-criteria based fuzzy rules, which is found to outperform decisions based on conventional decision rules (such as EDD, FCFS). Simulation experiments have been conducted by most of the authors, while other limit themselves to information models and specifications.

Although many distributed frameworks have been proposed in the past, they are limited by the inability to learn from past decisions, and by the absence of a global perspective in making local decisions. As a result, the myopic local decisions do not control the shop in an optimal manner. In this paper, we present a distributed adaptive control system composed of distributed entities (agents) that will perform adaptive scheduling and schedule execution. The novelty in our approach is in the development of a supervisory component, which passively monitors the proceedings of the system and gains knowledge about the global performance of the shop.

2. PROPOSED SHOP FLOOR CONTROL AND SCHEDULING FRAMEWORK

A framework consisting of interacting, distributed entities (agents) is presented as a means to schedule and control the material and information flow in a shop. Agents are autonomous, independent entities that communicate and negotiate with each other in order to perform the desired task. The proposed framework (Figure 1A) consists of four types of agents: Part agent (PA), Part Manager agent (PMA), Machine agent (MA) and Supervisor agent (SA). When a new part enters the system, the PMA assigns a PA to the part. The PA guides the part through the various operations to be performed, via interactions with the MA. Data with regards to the movement of the part is submitted to the PMA before the part exits the system. The SA interacts with the MA's, PMA and the PA's and fine tunes the functioning of the agents based on the learning from the performance of the shop.

The agents are formally represented using set notation as follows:

$$A = \{S, E, M, R, D\} \quad (1)$$

where

- S is the set of all possible states of the agent. The process plan, the operations completed, the remaining operations, etc, describe the state of the PA. The set-up times, current set-up, number of parts in the input and output buffers, etc, describe the state of the MA. The different part types, routing information of the parts, previous routings of all the parts of all types, number of parts of each part type in the system, etc, describe the state of the PMA. The state of the SA is described by the number and location of the parts of each part type in the system, the status of the machines, and buffers.
- E is the set of possible environments of the agents. This would contain the information obtained via interaction with the other agents. For example, the PA's would contain the bid information of each MA's.
- M is the set of all managerial attributes. The PA and MA include information on their allocated budgets, priority, goals, etc. The SA includes information regarding the global objectives and expected performance.
- D is the set of all possible decisions that can be taken by the agent, and

R is the set of all possible decision rules that map S, E and M onto D. Each rule is associated with a weight which is modified based on the learning from past decisions.

Each part is associated with an agent, namely the PA. The PA contains all the data about the part (e.g., process plans), the managerial information (e.g., local goals, budget allocated), and the decision-making rules for negotiations. The PA's do not communicate with one another, they communicate only with MA's. Each machine in the shop is associated with a MA. The MA contains the data about the machine, managerial information, and the decision-making rules. The negotiation protocol between the PA and MA is illustrated in Figure 1B. It is seen that the protocol is initiated by the PA, who sends the task announcements to multiple MA's. Each MA then constructs and submits its bid. The PA, then, evaluates the bids and selects the machine to carry out the operation.

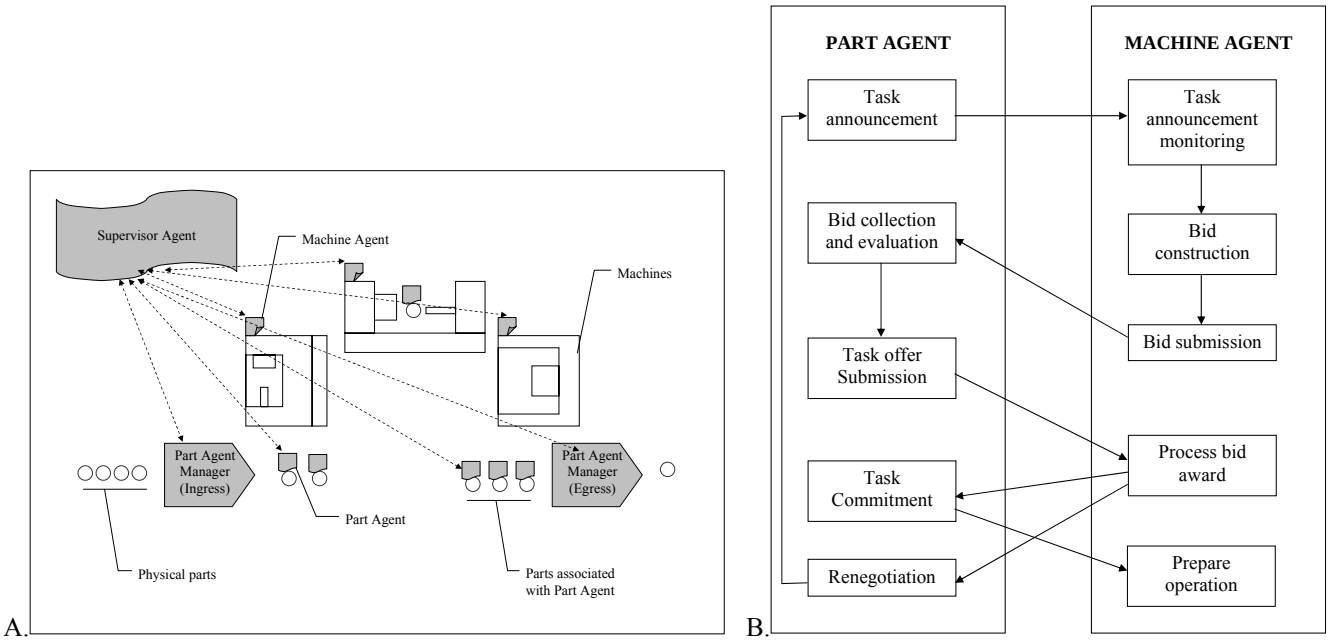

Figure 1A: Proposed architecture overview; 1B: Model of negotiation between PA & MA (Source: Cavalieri et al., 2000)

The decision-making rules of the PA and MA are based on the immediate environment conditions. The relationship between the local decisions and the global performance is vague and uncertain. In the proposed framework, in order to overcome the shortcomings, the use of a Part Manager Agent (PMA) and a Supervisor Agent (SA) is proposed. The PMA assigns a clone of the PA to each part when the part enters the system. When the part exits the system, typically the data collected or the learning of the part is lost. In the current framework, provision has been provided for the PMA to collect and maintain a record of the data collected from the completed PA's. The role of PMA is passive in the sense that it does not influence the proceedings within the shop.

The learning and a global perspective of the system are provided by the Supervisor Agent. The SA interacts with the MA's, PA's and PMA to gather information regarding the current status of the shop. Managerial information such as the overall goal of the scheduling system is mapped with the local decision rules of the lower level agents using artificial intelligence techniques (e.g., neural networks, fuzzy inference systems) in bid to arrive at a global optimum. The SA fine tunes the decision rules of the lower level agents to make the system more efficient.

3. DESIGN ISSUES OF CONTROL AND SCHEDULING COMPONENTS

3.1 Assignment of part and machine

The problem of assigning parts to a machine is separated into two parts, part selection and machine selection. The part-selection decision arises when a more than one part compete for the same machine. The machine selection decision arises when more than one machine can process the same part, or the next operation on the part at a machine requires a different

setup than the current operation. While the former is obvious, the latter is not. For example, a part has two operations to be performed. The operations can be performed either in the same machine with different set-up for each operation or in two different machines. If the second setup takes a longer time (compared with material handling time), then the part can be sent to the next machine. If the second setup is short or the next machine is busy, then the part can be processed in the same machine itself. In all cases, the next machine to be visited by the part must be determined - sometimes there will be one choice; sometimes there will be multiple choices.

3.2 Negotiation procedures

Interaction between part and machine agents is required to schedule and route the part through the system. Let $O = \{o_1 \ldots o_n\}$ represent the set of all possible operations that are performed in the shop. Let $P_i = \{p_{i1} \ldots p_{im}\}$ represent the set of all the operations that is to be performed on part i ($P_i \subset O_i$). Let $Q_j = \{q_{j1} \ldots q_{jm}\}$ represent the set of all the operations that is to be performed by machine j ($Q_j \subset O_i$). Jobs are scheduled in machines in advance for the next λ time period. The time period is segmented into λ time slots of unit length each. The value of λ can be different for different machines. When a part enters the system, it initiates the negotiations by sending announcements to all the machines capable of performing its first operation (p_{i1}). The information the part sends includes the part type, operation to be performed, time slots requested, priority, price offered, and earliest and latest possible start time of the part. The machine on receiving the announcement checks its current schedule for the next λ time period and allocates the part the requested time slots, if desired (Figure 2). The price offered by the part is in inverse proportion to the start time of the operation scheduled on the machine. That is, if the part is scheduled at the earliest possible time requested, the machine gets the maximum price; if the part is scheduled at the latest possible time requested, the machine gets the minimum price. Upon confirmation from the machine, the part schedules the next operation that needs to be performed on the same or another machine. The earliest possible start time of the next operation is the estimated completion time of the previous operation, and the latest possible start time of the next operation is the completion time of the previous operation assuming it starts at the latest possible time and transportation delays are negligible.

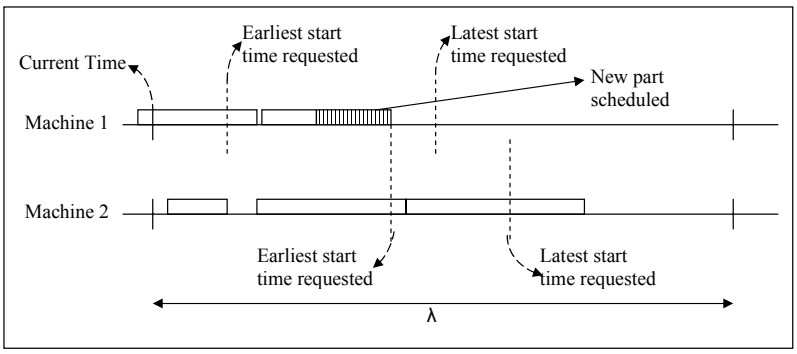

Figure 2: Illustration of the scheduling procedure

The generation of a complete schedule is compromised by various conditions of the shop, which necessitates negotiations. Inability of the machine to schedule the operation within the part's requested time frame is handled in one of the following ways: 1) the part can schedule the same operation on a different machine, 2) the part can provide a different time frame within which the machine would try to schedule the part, or 3) the machine can change its schedule by 'removing' currently scheduled jobs to make space for the new job. The second option is achieved via negotiations between the PA and the MA. The new time frame is determined by the part with respect to its current priorities and goals. In the third option, the machine can 'remove' an existing scheduled part from its current position in the schedule. The part that is removed will try to schedule itself in the same or other machines via negotiations procedures. All the future schedules of the removed part that are disrupted will be rescheduled.

When multiple parts request for time slots within the same time period, the MA bases its decision on the priority of the part, price offered, and the machine's current schedule. The part, after every round of failed negotiation to schedule its next operation, increases its priority or price, or modifies the time frame as required. This is to ensure that the part will not get stuck in the system.

3.3 Blocking and deadlock

The next issues of concerns are blocking and deadlock. Blocking occurs when a part has finished processing in the current machine, but can't vacate that machine until the downstream machine becomes empty and idle. Blocking is a temporary state of the system which can resolve itself over time. In contrast, deadlocking is a catastrophic event, causing the system to come to a stand still. Deadlock occurs when, for example, part 1 currently done processing in machine A needs to visit machine B next; and part 2 currently done processing in machine B needs to visit machine A next. This is a typical case of deadlock, as both parts wait for their next machine to be free to vacate the current machine. Deadlock can be avoided by,

(1) Coming up with a deadlock free schedule: This requires the use of a higher level controller with a global perspective and look-ahead strategy. The use of local autonomous agents does not permit such a scheme.
(2) Allowing only one part into the system/cell: This method will never allow deadlock to arise as the next machine to visit is always free. However it is neither efficient nor optimal.
(3) The use of an intermediate buffer: The part is transferred to the intermediate buffer when it has completed processing in the current machine and the next machine to visit is busy. Deadlock can still arise if the buffer size is too small. Hence, a buffer of sufficiently large capacity is assumed to be present.
(4) The MA makes uses the information about the previous operation and the next operation of the part in making its decision. This provides a feasible solution to deal with deadlock.

The various algorithms handling the deadlock will be evaluated and the most suitable one will be incorporated into the proposed framework.

3.4 Handling Disturbances

The next issue of concern is in handling disturbances. The proposed framework is intended to be robust to handle all types of disturbances. The common sources of disturbances are machine breakdown, rush job arrivals, change in job priority, and process time variation. When a machine fails, the currently scheduled jobs will be 'removed'. These removed jobs will then try to schedule themselves as described in Section 3.2. The movement of rush jobs in the system will be handled inherently by assigning higher priority attribute and increased allotted funds to those parts.

3.5 Decomposing the global objective to agent objectives

The control system of the shop floor is expected to optimize the global performance, even when the shop is subject to disturbances. In the proposed distributed multi-agent system, the decision power is to be distributed among the agents. The agents will then try to optimize their own local objectives which will in turn maintain the global performance within an acceptable level. The optimal scheduling problem for the global objective is often formulated as a general optimization problem with constraints, as shown in Equation 2. The decision variables $x_1, x_2, x_3, \ldots x_4$ are selected such that the objective function is either maximized or minimized, where each x_i is restricted by some constraints.

$$\begin{aligned} &\text{Optimize:} && f\left(x_1, x_2, \ldots, x_n\right) \\ &\text{subject to:} && g_j\left(x_1, x_2, \ldots, x_n\right) \le b_j,\ j = 1, 2, \ldots, m \\ &\text{where:} && x_i \in X,\ i = 1, 2, \ldots, n \end{aligned} \tag{2}$$

One of the algorithms that can be used to distribute the above function to each agent is the near optimal bounds Lagrangian relaxation technique. Lagrangian multiplier can be used to relax some of the constraints of Equation (2) by adding them in the objective function. For example, in Equation (2), any k of the m constraints (where $0 \le k \le m$) can be expressed in the objective function, resulting in an optimization problem of the form shown in Equation (3). In this formulation, the Lagrangian multiplier λ_j ends up being function of the decision variables, and the decision variable x_j ends up being function of the Lagrangian multiplier λ_j's. This optimization problem can be solved iteratively, by solving the dual problem. The optimum value of objective function is bounded by the optimal solution to the dual problem.

Subsequently, Equation (3) can be decomposed into the sub-problem of part, where the Lagrangian multipliers correspond to each of the machine. Thus the optimization of global objective can be distributed to PA's and MA's. Iterative negotiation of the agents will solve for the global objective (Baker, 1998). Each round of negotiation among the

agents will now be able to check as to how far it is from the bound on optimality, so the global performance can be maintained.

$$\begin{aligned} &\text{Optimize:} && f\left(x_1, x_2, ..., x_n\right) + \sum \lambda_j \left[g_j\left(x_1, x_2, ..., x_n\right) - b_j \right], j = 1, 2, ..., k \\ &\text{subject to:} && g_j\left(x_1, x_2, ..., x_n\right) \le b_j, j = k+1, ..., m \\ &\text{where:} && x_i \in X,\ i = 1, 2, ..., n \end{aligned} \tag{3}$$

4. CONCLUSION

In this paper, a framework for a distributed adaptive control system for a general shop floor has been presented. The control components are implemented as agents: part agent, machine agent, part manager agent and the supervisor agent. The novelty in our approach is in the development of a supervisory component, which passively monitors the proceedings of the system and gains knowledge about the global performance of the shop. Various design issues such as part-machine assignment, deadlock avoidance, disturbance handling, and the decomposition of the global objectives into multiple local objectives have been discussed. Future work will focus on formally defining the negotiation procedures between the different agents, including 1) what information/messages are exchanged, 2) when the messages are exchanged, and 3) what are the states of the agents. Forthcoming work will also be directed towards defining the decision rules of the part agent and machine agent and relating the local rules with the global performance of the system.

5. REFERENCES

1. Adacher, L., Agnetis, A., and Meloni, C. (2000). Autonomous agents architectures and algorithms in flexible manufacturing systems. IIE Transactions, 32: 941-951.

2. Baker, A.D. (1998). A survey of factory control algorithms which can be implemented in a multi-agent heterarchy: Dispatching, Scheduling and Pull. Journal of Manufacturing Systems, 17(4): 297-320.

3. Cavalieri, S., Garetti, M., Macchi, M., and Taisch, M. (2000). An experimental benchmarking of two multi-agent architectures for production scheduling and control. Computers in Industry, 43: 139-152.

4. Dilts, D.M., Boyd, N.P., and Whorms, H.H. (1991). The Evolution of Control Architectures for Automated Manufacturing Systems. Journal of Manufacturing Systems, 10(1): 79-93.

5. Macchiaroli, R., and Riemma, S. (2002). A negotiation scheme for autonomous agents in job shop scheduling. International Journal of Computer Integrated Manufacturing, 15(3): 222-232.

6. Maione, B., and Naso, D. (2001). Evolutionary adaptation of dispatching agents in heterarchical manufacturing systems. International Journal of Production Research, 39(7): 1481-1503.

7. Mehta, S.V., and Uzsoy, R.M. (1998). Predictable scheduling of a job shop subject to breakdowns. IEEE Transactions on Robotics and Automation, 14(3): 365-378.

8. Odrey, N.G., and Mejia, G. (2003). A reconfigurable multi-agent system architecture for error recovery in production systems. Robotics and Computer Integrated Manufacturing, 19: 35-43.

9. Park, S.C., Raman, N., and Shaw, M.J. (1997). Adaptive scheduling in dynamic flexible manufacturing systems: a dynamic rule selection approach. IEEE Transactions on Robotics and Automation, 13(4): 486-502.

10. Sabunchuoglu, I., and Bayiz, M., (2000). Analysis of reactive scheduling problems in a job shop environment. European Journal of Operational Research, 126: 567-586.

11. Solberg, J.J., and Lin, G.Y.L. (1992). Integrated shop floor control using autonomous agents. IIE Transactions. 24(3): 57-71.