Int J Adv Manuf Technol (2003) 21:980–984

The International Journal of
Advanced
Manufacturing
Technology

A Multi-Agent-Based Agile Scheduling Model for a Virtual Manufacturing Environment

Z. D. Zhou[1], H. H. Wang[1], Y. P. Chen[1], W. Ai, S. K. Ong[2], J. Y. H. Fuh[2] and A. Y. C. Nee[2]

[1]School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, 430074, P R China;
[2]Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260

In order to develop a multi-agent-based scheduling system for a virtual manufacturing environment, it is necessary to build various functional agents for all the resources and an agent manager to improve the scheduling agility. In this paper, a hybrid hierarchical model for agile job scheduling in a virtual workshop environment is proposed. The operations of the virtual manufacturing units, the negotiation processes and protocols among the agents are described in detail. Furthermore, forward and backward searching methods supported by the negotiation protocol are proposed for task assignment.

Keywords: Multi-agent-based scheduling; Negotiation protocol; Virtual manufacturing

1. Introduction

As one of the advanced manufacturing technologies of the 21st century, agile manufacturing has been studied extensively for customer-oriented production. While virtual manufacturing is an important means of realising agile manufacturing, scheduling is a bottleneck in achieving high productivity, flexibility, and reliability. To improve the agility, flexibility, and reliability of a manufacturing system, research and application of optimised scheduling methods have become the foundation of advanced manufacturing technologies [1–8].

Multi-agent-based scheduling is a new intelligent scheduling method based on the theories of multi-agent system (MAS) and distributed artificial intelligence (DAI). Multi-agent-based scheduling can dynamically and flexibly schedule manufacturing processes and rapidly respond to market demands by means of cooperation and coordination among the agents. In this work, a multi-agent-based scheduling system is developed for a virtual manufacturing environment. Different functional agents are built for all the resources in a manufacturing system, and a new hybrid hierarchical model for agile job scheduling in a virtual workshop environment is proposed. The negotiation processes and protocols among the agents are described in detail.

Correspondence and offprint requests to: Dr S. K. Ong, Department of Mechanical Engineering, National University of Singapore, Singapore 119260. E-mail: mpeongsk@nus.edu.sg
Received 12 March 2002
Accepted 13 May 2002

2. Concepts of Agents and MAS

2.1 Concept of Agents

An agent is an autonomous software entity with sovereignty, interactivity, and responsibility. Each agent performs its own tasks and completes jobs for the whole system through communication and coordination among the other agents. As shown in Fig. 1, the basic structure of an agent consists of the following three modules, namely the manager module, message processor, and communication protocol.

Manager module. This is responsible for interacting with other agents, and executing the assigned tasks with a manager unit, a knowledge base, a method base consisting of the required operations, a database, and an agenda, which is a list of the scheduled tasks.

Message processor. This module stores and processes the messages received and the tasks assigned from the users or other agents.

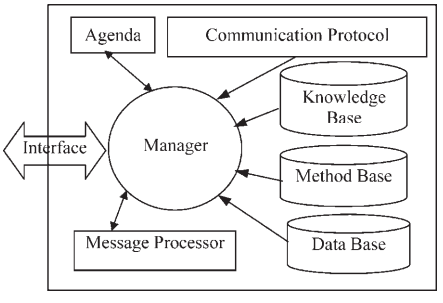

Fig. 1. Basic structure of an agent.

Communication protocol. This guarantees the continuity and compatibility of communication between the agents using a unified communication language and protocol.

Consisting of several intelligent agents, an MAS is a computer system that has the ability to complete jobs through the cooperation among these autonomous agents. Each agent is capable of acting autonomously, cooperatively, and effectively to achieve a common goal. MAS has been used in enterprises to solve difficult problems in distributed scheduling, dynamic scheduling and flexible scheduling, and particularly, for handling unforeseen circumstances.

2.2 Equipment Agents in a Virtual Workshop

As there are many types of equipment resource in a virtual workshop (e.g. NC machine tools, AGVs (automated guided vehicles), manipulators) it is necessary to build different agents for all these resources. All the state set, the capability set, the method set, and the rule set of the virtual manufacturing resources are determined by the agent-wrapping processes. This can be described as a hexahydric group as follows:

$$Equip\_Agent_i = (AgentId_i, State_i, Capability_i, Method_i, Rule_i, Constraint_i) \quad (1)$$

where, $AgentId_i$ is the exclusive agent identifier, $State_i$ is the state set, $Capability_i$ is the capability set that consists of all the functions executed by $Equip\_Agent_i$, $Method_i$ is the method set that comprises all the service processes, $Rule_i$ is the knowledge base of $Equip\_Agent_i$, and $Constraint_i$ contains some related constraints related to the i^{th} equipment. These variables can be expressed as follows:

$$\begin{aligned} State_i &= (Data_{1i}, Data_{2i} \ldots, Data_{ni} \\ Capability_i &= (Capa_{1i}, Capa_{2i}, \ldots, Capa_{mi}) \\ Method_i &= (action_{1i}, action_{2i}, \ldots, action_{ki}) \\ Rule_i &= (rule_{1i}, rule_{2i} \ldots, rule_{ji}) \end{aligned}$$

Where n, m, k, and j are the element counters of $State_i$, $Capability_i$, $Method_i$ and $Rule_i$, respectively.

To implement these equipment agents, the agent-wrapping process should first be analysed. Equipment resources are encapsulated as equipment agents to allow the equipment to function autonomously. As different manufacturing resources have different behaviours and characteristics, each manufacturing resource agent must be defined individually. The wrapping process usually involves two steps:

1. Building an adaptive layer around the existing controllers to transform them into normalised servers.
2. Building an agent manager with functionalities similar to that of the resource being wrapped.

Figure 2 shows the structure of a virtual equipment resource agent. The server, acting as a wrapped machine controller, controls the tasks execution. The manager handles the agenda and offers the capabilities of the server to the agent community. While the manager continues to negotiate with the agent community for the server's capabilities, the server continues to execute its contracted tasks. Thus, this architecture can speed up the entire scheduling process.

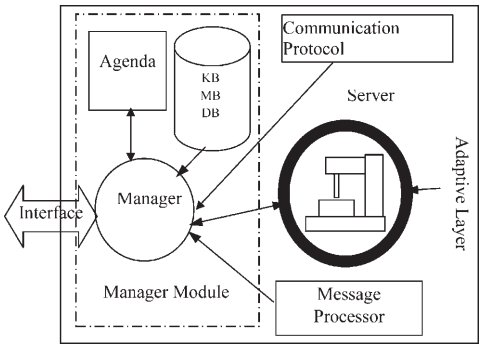

Fig. 2. Virtual equipment resource agent.

An equipment agent, such as an agent of a robot arm, can be implemented using the following formal definition:

Agent *<robot-arm>*
  Private *data1, data2, data3, …*
  Knowledge-base *rule1, rule2, rule3, …*
  Process *<process-name1>*
    On *<event1>* **Do** *<action1>* **at Priority** *<priority1>*
  Process *<process-name2>*
    On *<event2>* **Do** *<action2>* **at Priority** *<priority2>*
  …
  Action *<action1>*
  Action *<action2>*
  …
  Processor *<processor-address>*
End

3. Architecture of the Multi-Agent-Based Agile Scheduling System

An agile scheduling system should have the following four capabilities:

1. Dynamically reacting to the unforeseen events that might happen in the current schedule, i.e. equipment breakdown or urgent jobs.
2. Considering the entire production resources, which are distributed beyond the traditional physical boundaries of shop-floors in a virtual enterprise.
3. Recombining the equipment resources during scheduling.
4. Using the process plan information by interfacing to a CIM (computer integrated manufacturing) information system.

In a multi-agent system, a distributed autonomous scheduling model replaces the centralised control model and a negotiation process for decision-making is used instead of pre-planned processes. In addition, there is a concurrent execution of tasks instead of the usual sequential processing of tasks, and different problem solvers in the same environment are used in place of a fixed problem solver. Using these means, the multi-agent-based modelling method significantly improves the agility and flexibility of scheduling. Therefore, it has become a representative method in the development of complex, flexible, and intelligent scheduling systems [3,5].

3.1 Hybrid Hierarchical Architecture

In MASs, there are two types of architecture. The first is a distributed architecture that is used only in small-scale systems. If the scale of the system is large, its architecture will become very complex, and the agent communications will be very complicated for the large number of agents in spite of the ability of the distributed architecture to achieve high autonomy. The second is a united architecture with mediators to aggregate the agents into agent groups and coordinate the communications and operations among the agents. Within a group, the mediator coordinates the activities of the agents and represents the group in the communications with other mediators and agents, while the agents in a group can only communicate with the mediator of the group. In this way, the communication process becomes simpler compared to that of the distributed architecture. However, the communication to the supervisor node becomes overloaded and the system is more sensitive to malfunctioning. In this paper, the two types of architecture are integrated and a hybrid hierarchical architecture is proposed for the scheduling system (Fig. 3).

In this model, there are three layers, namely the scheduling manager agent layer, task agent layer and production resource agent layer. The most important characteristic of this hybrid hierarchical architecture is that the upper layer agent (i.e. the scheduling manager agent) manages the lower one (i.e. the task agent), while each agent in the same layer can communicate with each other without the supervision of the agents in the upper layer. Among the agents, the KQML (knowledge query management language)-based negotiation protocol (NP) is used to support the communication. The manufacturing automation protocol (MAP) is used to enable communications between the production resource agents and the resources while the STEP (standard of exchange product data) protocol is used to communicate with the CIM information system.

Global scheduling is executed by the scheduling manager agent, which acts as a bridge between the system and other systems. After receiving orders from the users and acquiring the manufacturing process planning information from the CIM information system, the scheduling manager agent will evaluate the orders by considering its state and the assigned load. If it is able to execute the orders, it will transform the orders into production task agents, and initiate a task agent to execute a specified task; otherwise, it will communicate with some cooperative partners to subcontract a part of the orders to them. It will also be responsible for dealing with dynamic changes in the task conditions (e.g. a change of the duration of a task) and coordinate the activities of the task agents based on the execution plan and the status of every task agent. In this way, human and machine intelligence are integrated in a scheduling manager agent to realise the information transformation from users to tasks.

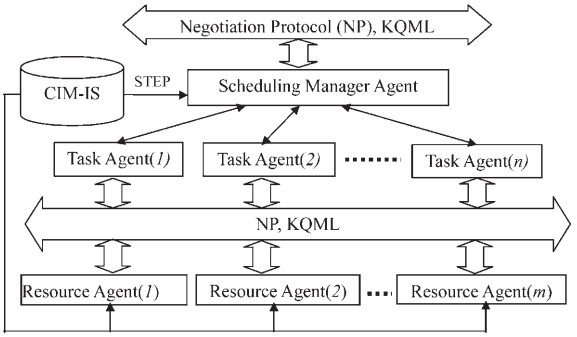

Fig. 3. Hybrid hierarchical architecture of a multi-agent-based scheduling system.

A task agent is launched temporarily by the scheduling manager agent for a special task. Once started, it directly negotiates with suitable resource agents to select an appropriate resource for a specified work procedure. Each task agent has a life-span from birth to death. During its existence, a task agent is responsible for monitoring the progress of the work. Hence, it has a control system with a monitor–detect–react loop for process monitoring and control so as to realise the information transformation from production tasks to working procedures.

A production resource agent represents the current state of a resource, which includes the status and activities of the resource, and negotiates with other agents for the use of other production resources. It is responsible for monitoring the whole executing progress of a working procedure. Furthermore, it can resolve the conflicts of simultaneously employing the same resource from two resource agents by consulting with other agents. It can also subcontract part of the tasks to other resource agents if it cannot complete such a task on schedule owing to some unforeseen reasons. In this way, it is able to solve problems without communicating with the manager agent.

3.2 Negotiation Protocol

Conflict resolution is a major concern in traditional scheduling methods, because some incidents, e.g. time conflicts, storage capacity conflicts, or technical faults, will happen in the scheduling process or during the scheduling execution. These incidents may be caused by process planning, scheduling, or monitoring of task execution. Therefore, a negotiation protocol should be used to assign tasks to agents and resolve conflicts. To assign tasks to production resource agents based on time order, as shown in Fig. 4, three basic steps have to be performed. First, a task agent releases a task to some appropriate resource agents, These resource agents will bid for the task. Finally, the task agent will select the best agent to execute the task. The negotiation process and the information communicated among the agents are given in Table 1.

When the resource agents receive the resource request announcement message (RRAM) of task k, they will exchange the list of interval messages (LIM). The resource bid message (RBM) of each resource agent will be obtained at the end of the negotiation. This algorithm is described next.

In order to achieve the objective of the possible shortest production time, e.g. *Min(Op(n).endTime-Op(1).startTime)*, the negotiation of a task agent and the resource agents consists of two phases, namely, the forward-searching phase and the backward searching phase.

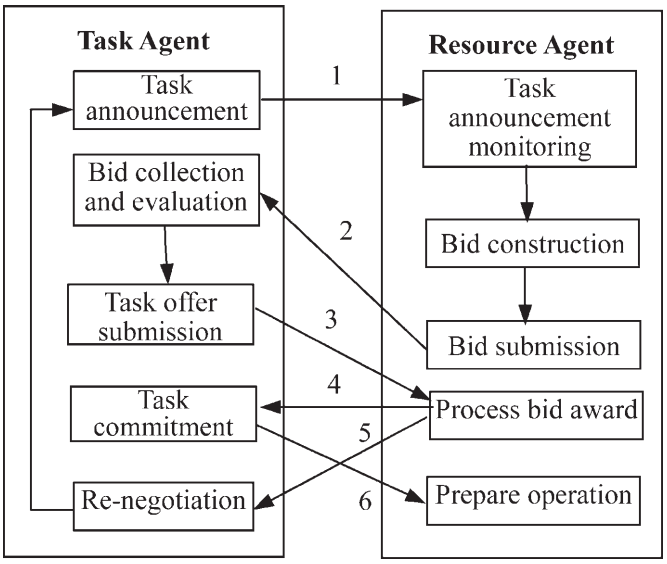

Fig. 4. The task negotiation process. 1. Task available. 2. Bid submission. 3. Bid award. 4. Bid confirmation. 5. Bid rejection. 6. Task ready.

3.2.1 Forward Searching Phase

Supposing that there are n operations registered in the agenda list of a resource agent i (named as $Res(i)$), they can be expressed by their required time interval as an engaged time interval set as follows.

$$Op(Res(i)) = \bigcup_{j=1}^{n} SelTimeInterval_{i,j} \tag{2}$$

where, $SelTimeInterval = \{startTime,\ endTime\}$.
The free time interval set of $Res(i)$ within the time window (TW) is

$$Free(Res(i)) = \overline{Op(Res(i))} \cap TW \tag{3}$$

Once a resource agent receives a RRAM message, it analyses the time intervals of its agenda for the requested operation. The resource agent, contacted to the operation(s) without predecessors, will start the forward influence phase. This phase means that the time intervals list for the next operation is influenced by the resulting list of the previous operation(s), and the free time intervals list of this resource is considered as the time window (TW) for the other operations before the requested operation.

Beginning with the first operation Op_1, according to the principle of earliest starting time, an operation marker MT_1^i is inserted in each of the appropriate resource agent $Res(i)$ ($i = 1$ to q, and q is the number of the appropriate resource agents), such that

$$\|MT_1^i\| = Duration\ (Op_1) \tag{4}$$

As a result, the engaged time interval set and the free time interval set of production agents can be calculated using Eqs (5) and (6), respectively.

$$Op(Res(i)) = Op(Res(i)) \cup MT_1^i \quad (i = 1,\ldots,q) \tag{5}$$

$$Free(Res(i)) = \overline{Op(Res(i))} \cap TW_1^i\ (i = 1, \ldots,q), TW_1^i = TW \tag{6}$$

The next operation Op_2 is considered. Owing to the ordinal requirement of the operations, the start point of TW_2^i should be moved behind the end point of MT_1^i. For Op_2, the steps above are repeated to the appropriate resource agents $Res(j)$ ($j = 1, \ldots,p$, and p is the number of appropriate resource agents) to calculate the engaged time interval set and the free time interval set of the production agents after scheduling the Op_2. In this way, the procedures are repeated until the last operation, and the resource engaged time interval sets of all the appropriate resource agents are obtained for task k.

3.2.2 Backward Searching Phase

The backward searching phase is a reverse searching process. Related to the resource engaged time interval set obtained from the forward searching phase, it starts with the last operation and selects one of the corresponding resource agents according to the principle of earliest ending time. The resource agent(s) of the last operation(s) will influence the intervals list of the previous operation(s) by sending the list of interval message (LIM). After the resource agents for the first operation have received the LIM message from the previous operations, the tasks of this phase are accomplished. After the execution of the two phases, different suitable equipment resources will be chosen for every operation of task k respectively.

4. Simulation Results

Supposing that there is a virtual workshop consisting of two working clusters, namely cluster A and cluster B to simulate the working efficiency of the above algorithm. It will take 10 min to transfer a workpiece between them. There are two machine tools (i.e. machine tool 1# and machine tool 2#) in cluster A and another two machine tools (i.e. machine tool 3# and machine tool 4#) in cluster B. All of them are ready to work and their job agendas are empty at the beginning. In addition, there are four workpieces and each of them will require five different operations in a specified order. Table 2 shows the alternative machines for processing the parts along with the respective processing times.

The related schedule of the example is obtained in the form of a Gantt chart, shown in Fig. 5, where, "P1" stands for "workpiece 1", "M1" stands for "machine tool 1#", and "1-5-3" means that the 5th operation of workpiece 1 is done on machine tool 3#. From the Gantt chart, it is shown by the simulation result that the multi-agent-based scheduling method is much simpler and more effective.

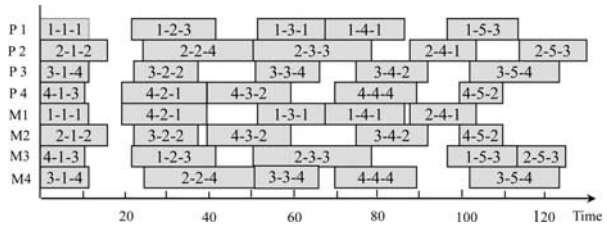

Fig. 5. Gantt chart of makespan.

Table 1. Information communicated among agents.

Information name	Information contents	Sender	Receiver
Task announcement message (TAM)	(Number of parts, item, time window, {task constraint}, task descriptor)	Manage agent	Task agent
Task process message (TPM)	(Procedure name, resource, duration, tools, components)	Manage agent	Task agent
Resource request announcement message (RRAM)	(Operation number, operation, time window, {task constraint}, task descriptor, {previous operation, {pre-resource}}, {next operation, {next-resource }})	Task agent	Resource agent
List of interval message (LIM)	(resource, operation, {time-interval}, task descriptor, direction)	Resource agent	Resource agent
Resource bid message (RBM)	(Resource, operation, {time-interval, free-time-till-end}, task descriptor)	Resource agent	Task agent
Resource select message (RSM)	(Operation, select-time-interval, task descriptor)	Task agent	Resource agent

Table 2. Processed workpieces.

	Operation 1	Operation 2	Operation 3	Operation 4	Operation 5
Workpiece 1	$1^{\#}$(12), $2^{\#}$(18)*	$3^{\#}$(20), $4^{\#}$(24)	$1^{\#}$(16)	$1^{\#}$(20), $2^{\#}$(18)	$3^{\#}$(15)
Workpiece 2	$1^{\#}$(20), $2^{\#}$(16)	$4^{\#}$(25)	$3^{\#}$(28)	$1^{\#}$(15), $2^{\#}$(19)	$3^{\#}$(21), $4^{\#}$(27)
Workpiece 3	$4^{\#}$(12), $3^{\#}$(14)	$2^{\#}$(16), $1^{\#}$(20)	$3^{\#}$(22), $4^{\#}$(16)	$1^{\#}$(19), $2^{\#}$(15)	$4^{\#}$(25)
Workpiece 4	$3^{\#}$(10), $4^{\#}$(15)	$1^{\#}$(20), $2^{\#}$(25)	$2^{\#}$(20), $3^{\#}$(18)	$4^{\#}$(20)	$1^{\#}$(15), $2^{\#}$(10)

*Where, $1^{\#}$(12) and $2^{\#}$(18) indicate that the first operation on workpiece 1 can be done on machine $1^{\#}$ within 12 min and also can be done on machine $2^{\#}$ within 18 min.

5. Conclusions

In this paper, an effective method for agile scheduling in a virtual workshop environment has been formulated based on a negotiation-based task allocation method. A hybrid MAS architecture is proposed to resolve the conflicts and respond to the unforeseen events. This multi-agent-based hybrid hierarchical architecture has been applied to agile scheduling in a virtual workshop. Using this architecture, the developed scheduling system will be much simpler and its reliability and robustness can be improved greatly.

Acknowledgements

The authors would like to acknowledge the support of funding from A*STAR, Agency for Science, Technology and Research for the China–Singapore Joint Research Programme between China and Singapore and the funding (Grant no. 5991076861) from the National Science Fund Committee (NSFC) of China, as well as the contributions from all collaborators of the mentioned projects. They would also like to thank Huazhong University of Science and Technology of China and the National University of Singapore in supporting this joint work.

References

1. R. J. Rabelo, L. M. Camarinha-Matos and H. Afsarmanesh, "Multi-agent-based agile scheduling", Robotics and Autonomous Systems, 27, pp. 15–28, 1999.
2. M. Ulieru, D, Norrie, R. Kremer and Weiming Shen, "A multi-resolution collaborative architecture for web-centric global manufacturing", Information Sciences, 127, pp. 3–21, 2000.
3. C. Lemaitre and C. B. Excelente, "Multi-agent network for cooperative work", Expert System with Application, 14, pp. 117–127, 1998.
4. V. G. Duffy and G. Salvendy, "Concurrent engineering and virtual reality for human resource planning", Computers in Industry, 42, pp. 109–125, 2000.
5. P. Sousa and C. Ramos, "A distributed architecture and negotiation protocol for scheduling in manufacturing systems", Computers in Industry, 38, pp. 103–113, 1999.
6. S. Cavalieri, M. Garetti, M. Macchi and M. Taisch, "An experimental benchmarking of two multi-agent architectures for production scheduling and control", Computers in Industry, 43, pp. 139–152, 2000.
7. Y. Yan, T. Kuphal and J. Bode, "Application of multi agent systems in project management", International Journal of Production Economics, 68, pp. 185–197, 2000.
8. P. C. Pendharkar, "A computational study on design and performance issues of multi-agent intelligent systems for dynamic scheduling environments", Expert Systems with Applications, 16, pp. 121–133, 1999.