Identification of Scheduling Problems for CSCW-based Shop Floor Control in Agile Manufacturing

Soohyun Cha, Hyunbo Cho, Mooyoung Jung

Department of Industrial Engineering, Pohang University of Science and Technology,
Pohang 790-784, Republic of Korea

ABSTRACT

Numerous solution methods for scheduling problems such as part dispatching problem, operation sequence problem have been suggested as a means to be embedded in hierarchical or centralized shop floor control. Under the preceding control philosophies, however, response to changes in the shop floor status is quite slow and timely decision is sometimes impossible. Moreover, the control software becomes too large and it is almost impossible to modify the control software when the configuration of the shop floor changes. In agile manufacturing which emerged recently to cope with quick response and easy modifiability when unexpected changes occur, a new control policy is needed. CSCW[Computer Supported Cooperative Work] based shop floor control casts a different view on scheduling problems. Decisions are made locally when requested and useful information is scattered among agents for its efficient use. Adaptation is easy because agents are plug compatible or portable. In this paper, scheduling problems occurring under CSCW based shop floor control are identified and characterized. Traditional scheduling problems are reviewed from the CSCW viewpoint. All the control entities involved in the shop floor can be found and used to defined agents. With these entities and CSCW concept, possible scheduling problems are identified.

Key words : CSCW-based shop floor control, shop floor entity, scheduling, autonomous agent.

I. Introduction

As industries go global and competition among them becomes more intense, manufacturing technologies become more important for survival. Markets require manufacturing of short lead time, production at low cost, and high product quality of course. To satisfy these market requirements, each shop floor must be able to quickly produce at a reasonable cost and be easily modified when the change in the configuration of the shop floor is needed. To this end, shop floor control system must also be able to support this shop requirements. In particular, scheduling under an appropriate control architecture is

crucial to meet those shop floor requirements. Under centralized or hierarchical control architectures, however, it is impossible to accommodate the requirements for modern manufacturing[5]. The manufacturing system under the traditional control architecture are operated in a top-down, centralized, sequential, and algorithmic manner, making the system difficult to be flexible, to modify, and slow to respond. Moreover, researches on shop floor scheduling were mainly on how to solve specific problems rather than on what to solve, thus resulting in the set of solution methods. But before a problem is solved, the problem should be first identified. Solution methods for the problem must be then sought.

In this paper, research effort is directed toward what to control instead of how. Specifically, this paper deals with the general scheduling problems in the machining shop. First, entities in the machining shop floor to be scheduled are defined and their attributes are identified. The entities are identified through extensive surveys on the existing shop floors , whether they are pilot plants or real manufacturing shops in industries. Defining these entities is the stepping stone for the scheduling problem generation. Second, scheduling under CSCW framework is defined. Third, scheduling problems under CSCW framework are generated using the entities and their attributes defined above. Most of the scheduling problems are defined in terms of the client-server relation.

II. Scheduling function under CSCW

It's difficult to define what scheduling is. Definitions on scheduling are slightly different from one another. But if a close look is taken at the role of scheduling defined until now, one common characteristic can be found, which is the role of decision making. Under any circumstances, scheduling must draw a decision when required. In the machining shop, thousands of events occur which require decision-making. The events include finish of part processing, arrival of several parts, machine break-down, and part rerouting, etc. When such events occur, scheduling is requested and it somehow has to draw a decision. Thus scheduling makes decisions when events associated with decision-making occur so that products can be produced within due-date. But in the real world, most of the events cannot be predicted. Certain events may be predicted. In that case, decision does not even have to be made, but instead, things will work as originally planned. On the contrary, most of the events are unpredictable and how to draw decisions is not easy. Scheduling under traditional control architectures, however, is not adequate to cope with thousands of problems occurring in the shop floor because,

- response to events is quite slow, for a central decision maker has to make all the decisions,
- traditional control architectures are based on complete accuracy in information for problem solving, but usually complete information may not be available at the moment of need,
- it cannot recognize the change in the status of the system as the system constantly and quickly evolves,
- control software adopting traditional control architectures becomes too large in size and it is almost impossible to modify when the configuration in the shop floor is changed.

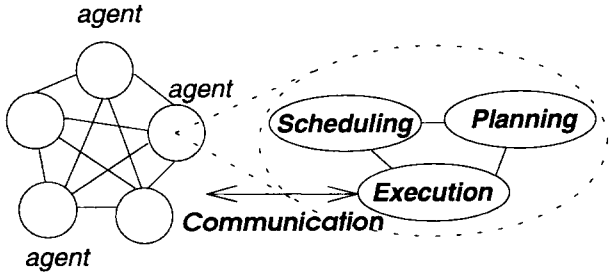
Thus, in this paper, scheduling under different control architecture is proposed. The architecture is based on CSCW [Computer Supported Cooperative Work]. Under CSCW, there is no central decision maker. Instead, decision and information are scattered around the shop floor[3]. When an event which

requires decision occurs, entities involved draw a decision cooperatively. Entities involved try to solve the problem using their local expertise. Even though completely accurate information is not available, the entities somehow draw a decision with the available information. Scheduling under CSCW seems to satisfy modern manufacturing requirements better than the traditional architectures because,

- decision problem can be quickly and locally solved because entities involved draw a decision as soon as the problem occurs, not requiring the central decision maker to draw a decision,
- local information and expertise can be fully utilized,
- changes in the configuration of the shop floor are easy, for entities with its own control software are simply added or removed without major change,
- scheduling system under CSCW can work without complete information, for CSCW architecture is based on functional accuracy rather than complete accuracy[4].

Due to the above advantages and others not cited here over the traditional architectures, scheduling under CSCW is being heavily studied world wide.

The control under CSCW can be operated by three functions, planning, scheduling, and execution[1].



[Figure 1. Composition of an agent]

Planning and scheduling functions can be classified into one functional unit, the decision making function. Execution function gets a message from planning and scheduling and takes actions according to the message. Another important role of the execution function is to accommodate the communication activities with other agents. The communication refers to not only usual data transfer but also the negotiation with other agents to draw a decision. Decision can be made either by a planning function or by a scheduling function. When the two functions need cooperation with some agents, execution function is requested and does some errand to negotiate for a decision. The distinction between planning and scheduling function is not clear. The purpose of the division is to divide burdens on the decision-making function into two so that each function can be easily implemented. One possible criteria of distinction is the frequency of the occurrence of the events. Scheduling function may be in charge of relatively frequent problems. Rigorous distinction will not be made here. In this paper, efforts are directed toward the identification of entities to be scheduled and scheduling problems in the machining shop.

III. Entities to be scheduled

Until now, almost all of the papers on scheduling have been concerned in how to solve problems. Each of the papers has been so obsessed with how to solve specific problems that they forgot what they were trying to schedule. But there do exist entities which comprise the shop floor and whose composition and attributes generate scheduling problems. Thus, the identification of the entities must be carried away before delving into solving scheduling problems. Identification of entities and attributes is useful in,

- defining what to be scheduled
- defining scheduling problems,
- simplifying scheduling problems which seem

[Tabke 1] Entities in the machining shop floor

Entity	Mobility	Flow Direction	Flow Pattern	Perishability	Capacity	Ownership	active/passive	Processing Capacity
Batch	mobile/fixed	uni/multi	random/static	Y/N	N/A	shared	N/A	uni/multi
Machine	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared/assigned	active	uni/multi
Part	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared	N/A	N/A
Material Handler	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared/assigned	active/passive	uni/multi
Tool	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared/assigned	N/A	uni/multi
Jig & Fixture	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared/assigned	N/A	uni/multi
Workspace / Path	fixed	N/A	N/A	N	limited/unlimited	shared	passive	uni/multi
Buffer Storage	fixed	N/A	N/A	N	limited/unlimited	shared/assigned	passive	uni/multi
Attendant	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared/assigned	N/A	uni/multi
Pallet	mobile/fixed	uni/multi	random/static	Y/N	limited/unlimited	shared/assigned	active	uni/multi

chaotic at first sight.

The entities with their attributes are proposed in Table.1 which is put at the end of this paper. The table is an updated version from the one by H. Cho in his dissertation for Ph.D[1]. The table was constructed through extensive survey on shop floors realized throughout the world. The shop floor includes those from research facilities as well as those in industries. Here are brief explanations on the entities and their attributes identified.

Entities

- Batch
- Machine
- Part
- Material Handler
- Tool
- Jig & Fixture
- Workspace or Path
- Buffer Storage
- Attendant
- Pallet

Attributes

- ***Mobility*** : This attribute indicates whether entities are movable or not. If an entity is mobile, the entity needs to be assigned to one of the fixed entities. For example, in a ship building problem, the part is fixed and machines are mobile. Hence, the flow characteristics of machines need to be addressed. Tools and fixtures can be also mobile or fixed. If tools are fixed to a machine, the machine has ownership of the tools and the tools need not be scheduled.
- ***Flow Pattern*** : This attribute indicates whether the next transition of entities is random or static. The random flow of entities makes a great impact on scheduling problems. For example, if a part is mobile and its flow pattern is random, the operation sequence graph for the part contains SPLIT-AND and JOIN-AND type nodes.
- ***Flow Direction*** : This attribute indicates whether all of the entities have the same direction or not. The property shows the interaction between two entities of the same class if they are mobile. If the flow pattern and the flow direction of entities are static and uni-directional, respectively, the flow of entities is easy to control.
- ***Perishability*** : This attribute indicates whether entities are perishable or not. If entities are perishable, entity wear must be considered as related criteria. For example, if a tool is perishable and shared and the remaining tool life is less than a specified value, machining parameters must be adjusted since tool wear depends on many of these parameters.
- ***Capacity*** : This attribute indicates whether entities are limited or not. Some entities(clients) need a set of related operations whose achievement requires other entities(servers). If the entities are limited, the clients will compete for time on them.
- ***Ownership*** : This attribute indicates whether entities(servers) are assigned to specific entities or shared by other entities(clients). The shared servers can be requested by several clients. Scheduling is not relevant unless the servers are shared.
- ***Passive/active*** : This attribute indicates whether entities are passive to the requests of other entities or not. Consider a number of types of material handlers. Direct-address material, such as robots, AGVs, hoists, cranes, need to make a

decision of the next destination after use. Indirect-address material handlers, such as conveyors, keep moving and are passively used by other entities.

- ***Processing Capacity*** : This attribute indicates whether a server can serve requests from many clients at the same time. If the server can serve, the server must decide whether to start serving or to wait after accepting some clients.

Of course, the entities and attributes defined above are not complete. There may exist other entities and attributes which are not identified here. But almost all of the machining shop floors surveyed commonly have these entities, and with the entities and attributes above, some important scheduling problems can be defined.

IV. Scheduling Problems

In this section scheduling problems are identified. The scheduling problems can be generated using the entities and their attributes defined in previous section. Client-server model is used to define the scheduling problems. Any entity defined in [Table.1] may become client or server, depending on how it can be used. A specific entity(client) requests other entities(server). Both client and server may belong to the same entity class. For example, a part may request a machine, tool, fixture, and attendant simultaneously in order to be machined. In the product assembly line, a specific part may also request other parts in order to be assembled according to BOM. Based on the client-server model, scheduling problems are defined and investigated.

i) ***Part Releasing problem***

:: Client many and server many

When some clients of the same entity class request servers belonging to one entity class, the scheduling function needs to decide the sequence of the clients. This problem arises when a new pallet arrives to a system and parts on the pallet must be released into the system.

ii) ***Part Dispatching problem***

:: Client many and server one

When some clients of the same class request one server, the server need to be granted to one of the clients. For example, many parts may request one processor at the same time. The scheduling function is then responsible for selecting one of the parts to be dispatched to the processor. Many parts also may request one tool or one fixture at the same time.

iii) ***Existence problem***

:: Client exists server none

This problem arises when clients request to be served but server does not exists. Then scheduling function must make decision whether to wait until server is ready or drop the clients and quit processing. The server may not exists due to break down, etc.

iv) ***Operation Sequence problem***

When a finished part has operation sequence alternatives specified by SPLIT-AND and JOIN-AND type nodes, a decision must be made regarding which operation should be processed first.

v) ***Part Buffering problem***

:: Client one and server many

This problem arises when one client requests one of several different server classes. The scheduling function is then responsible for granting one of the server classes to the client. A finished part on any machine may choose one of three options if the next machine is busy : going to the buffer storage, going to the material handler, or stay at the current location.

vi) ***Location problem***

:: Mobility attribute

It is commonly assumed that the flexibility of a FMS degrades when secondary resources including robots, tools, and fixtures are constrained(Hutchison and Khumawala[7]). Hence, secondary resources have been ignored when scheduling. However, it must be decided that the secondary resources must be scheduled after they're used.

vii) ***Bus driver problem***

:: Processing capacity attribute

If the processing capacity or a server is more than one, the server must decide whether to proceed to or wait to receive more clients. For example, a machine can server more than one clients, the machine has to decide whether to server the clients or wait till more clients arrive.

viii) ***Client Impairment problem***

Consider a situation where a part arrives to be served but it proves that the part is impaired. Then the corresponding server determines whether to proceed or drop the part and request other parts. That kind of a situation is called a client impairment problem.

ix) ***Server Impairment problem***

This problem arises when a part arrives to be served but the server cannot do its role because of break-down. Then scheduling function is responsible for the next action. This problem is different from the Existence problem because the Existence problem occurs before the part decides next destination. But the server impairment problem happens after a part is assigned to the specific server.

x) ***Parameter Modification problem***

Certain entities are perishable. Thus the scheduling function must operate considering the attribute. For example, if a tool being used has been used for so long a time that special care must be given in machining. The depth of cut, rotational speed may be modified in this case. The scheduling function is responsible for that.

xi) ***Batch Problem***

:: Caused by CSCW frame work

Even though FMS assumes the batch size of one, adequate size of the batch is still and will be important in the manufacturing of all sorts. But the definition of the size of the batch under CSCW is somewhat different from that of the traditional architectures. First, who will determine batch size is not clear. Second, what is the optimal batch size is not clear.

The scheduling problems described above are not complete problems that can be confronted in the shop floor. Above problems are the most general types of problems that occur in all the manufacturing shop floors. In addition to above problems, each agent may encounter its unique problems to be solved. Some of the problems can be solved without cooperation of other agents, while others need cooperation. If each agent which comprise the shop if equipped with the ability to solve its own problem and the ability to cooperate with other agents for the global objectives, the problems arising in the shop floor can be manipulated easily, reliably, and quickly.

However small a shop floor may be, there exist by far more problems that must be solved by a decision-making function than previously defined. But by considering the entities in the shop floor and their attributes, the scheduling problems of a specific shop floor can be easily generated and embedded in the shop

floor controller. As more of scheduling problems are identified, the shop floor controller can become generic and enrich its problem set.

IV. Conclusion

In this paper, entities in the shop floor are identified with their attributes to clarify what to schedule and to generate scheduling problems. Scheduling problems can be generated by considering the entities and their attributes. In investigating scheduling problems, a client-server model is used, which can be described according to object-oriented paradigm.

Once scheduling problems are defined, they are solved according to some rules or, if possible, efficient algorithms. Recently, multi pass simulation technique with look ahead feature is used in choosing the best possible rules.

It is imperative to additionally have a resource modeler for the automatic generation of scheduling problems. The resource modeler will capture entities existing in the shop floor. Various scheduling problems can then be generated automatically by the model constructed by the resource modeler.

Reference

[1] Cho, H., "An Intelligent Workstation Controller for Computer Integrated Manufacturing", Ph.D Dissertation, Texas A&M 1993

[2] Lee, K. C., Mansfield Jr., W., H., Sheth, A., P., "A Framework for Controlling Cooperative Agents", *IEEE Computer*, July 1993

[3] Hatvany, J., "Intelligence and Cooperation in Heterarchic Manufacturing Systems", *Robotics and Computer Integrated Manufacturing*, Vol.2, No.2, 1985

[4] Lesser, V. R. and Corkill, D. D., "Functionally Accurate, Cooperative Distributed Systems", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-11, No. 1, January 1981

[5] Dilts, D.M, et al, "The Evolution of Control Architectures for Automated Manufacturing Systems", *Journal of Manufacturing Systems*, Vol. 10, No. 1

[6] Gasser, L., "An Overview of DAI", Distributed Artificial Intelligence, Kluwer Academic Publishers, Edited Gasser, L., Dordrecht, Netherlands, 1992

[7] Hutchison, J. and Khumawala, B., "Scheduling random Flexible Manufacturing Systems with Dynamic Environments", *Journal of Operations Management* Vol. 9, No. 3, 1990