Association for Information Systems
AIS Electronic Library (AISeL)

PACIS 2001 Proceedings

Pacific Asia Conference on Information Systems (PACIS)

December 2001

An Agent for Selecting Optimal Order Set in EC Marketplace

Andrew Whinston
University of Texas

Hyung Choi
Dong- A University

Hyun-Soo Kim
Dong- A University

Young-Jae Park
Dong- A University

Byung-Joo Park
Dong- A University

An Agent for Selecting Optimal Order Set in EC Marketplace

Hyung Rim Choi, Hyun Soo Kim, Young Jae Park, Byung Joo Park
Department of MIS, Dong-A University, Pusan, Korea

Andrew B. Whinston
Center for Research in Electronic Commerce, University of Texas, Austin, USA

Abstract
Most small manufacturing companies rely on orders placed by buyers while selection and completion of these orders are closely linked to the load status of their production lines. The decision to accept an order or the selection of optimal order set, when orders exceed production capacity, critically depends on the production schedule. However, production scheduling is mainly performed by human experts so that most small manufacturers suffer from the inability to meet due dates or to make proper decision in accepting new orders. To address this problem, this paper develops an automatic agent that selects an optimal set of orders. The main engine of the selection agent is based on a typical job-shop scheduling model, formulating and implementing it as an Integer Program (IP) model. We also translate IP into Genetic Algorithm to address its NP-hard problem. We conclude with a suggestion for an agent architecture that tackles Web-based order selection problems.

Keywords: Intelligent Agent, Job-Shop Scheduling, Genetic Algorithm

1. Introduction

1.1 Backgrounds

The Internet-based electronic commerce is beginning to be recognized as a potential tool for small manufacturing companies to increase sales. However, small manufacturers cannot operate properly and optimally in the Internet environment due to their lack in money, personnel, and technology. For them to take advantage of the new tool, core technologies should be developed and improved in a way that supports electronic commerce by small manufacturing companies.
The changing digital environment requires small manufacturers to perform management

tasks, such as production scheduling, more rapidly and accurately in order to fulfill the requested due date of buyers in real time. Specially, molding industry market is order-based and buyer-oriented market. So it is very important to respond to their customer rapidly. Furthermore, because the production capacity of small manufacturer is very limited, they need to quickly select a set of optimal orders that will generate maximum profit when orders received from customer exceed their production capacity.

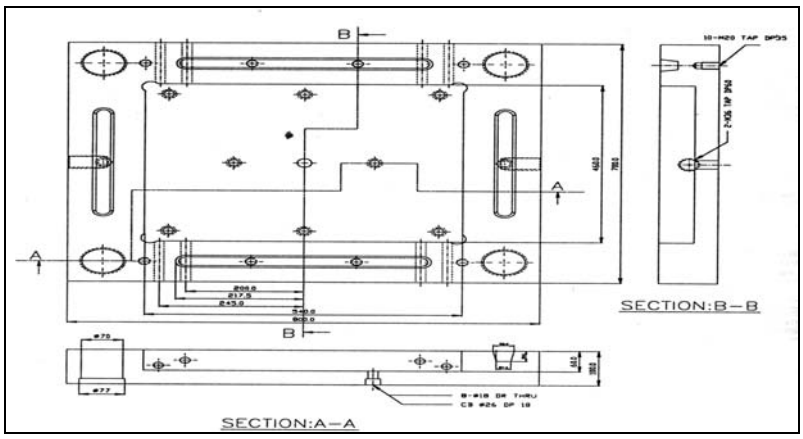
However, most small manufactures find it difficult to respond rapidly as they lack appropriate resources in personnel, funds and cost-effective technology. To cover these problems, agent that carries out production scheduling to respond rapidly and automatically without human intervention is needed.

In this paper, we present a methodology for rapidly selecting an optimal order set and provide a general architecture of selection agent for using it in the Internet environment. The methodology of selection agent developed in this paper is successfully applied to small injection-molding companies.

1.2 Research Scope

The target industry of our research is injection-mold manufacturer. A mold is a frame or shape composed of metal ingredients used for forming a shape by such processes as injection, press, casting, and forging. The injection molding is a production method acquiring geometry by rapidly stuffing high tempered melting material into a mold.

[Figure 1] A Drawing of Injection Mold, Divided Cavity Plate



Injection-molding products have various appearances and characteristics depending on products' shape and required functions. Machine and human labor are the most important resources in the injection-molding production. An injection-mold is produced

after series of operation each of which requiring a different machine. Milling, lathe, and drilling are some of the machines used for molding production. All of the operations have to be carried out in sequence; certain operation must be completed before moving on to next operation. Figure 1 shows the shape of a divided cavity plate as an example of injection molding.

While firms may invest in adequate machinery to support manufacturing activities, small and medium molding manufacturers in Korea suffer from poor production management. Especially, the problem seems to be in scheduling as most serious complaints from customers tend to be missed due dates. A confounding factor is the fact that most orders in the molding industry are order-based market and ad-hoc environment, thus these companies cannot rely on such a standardized process management as the ones used by mass producers. Profit-enhancing production management in this business must focus on meeting due dates and reducing costly overtime work.

This study aims to offer solutions for production management problems for small and medium molding manufacturers and to provide them with selection agent that can schedule and select proper orders with minimum computing facilities such as Internet-enabled client PCs. The selection agent-based scheduling analyzes such data as job operation sequence, processing time, due date, and profit of orders. When the analysis shows that molding manufacturers cannot produce all the orders, the selection agent automatically presents users with an optimal set of orders that generates the largest profit.

2. Literature Review

While intelligent agent has been a major part of continuing research in artificial intelligence, the field of intelligent agent began to rise as an independent research subject in the late 1980s. During the preceding decade, numerous international scientific conferences and symposia have been organized for artificial intelligence and the field has been extremely active as a number of agent-based products have been applied to real world commerce (Choi, 1997).

The precise definition of an intelligent agent varies. Franklin and Graesser postulated that this variance was due to the specificity of each model and agent system being implemented, each emphasizing only certain attributes that are salient to the tasks to be carried out (Franklin, Graesser, 1996). To arrive at a common definition, then, we need to review various attributes of an intelligent agent. These attributes are summarized below to differentiate them from other types of software applications (Nissen, 1995).

- Autonomy: The intelligent agent must have the capability to take actions leading to the completion of task(s) or objective(s) with or without impetus from the end-user. This relates to an element of independence of the agent.
- Communication ability: The intelligent agent must, in the course of achieving their objectives, access information from third party sources about the current "state" of the external environment. This requires an ability to communicate with the repositories of this information. Repositories may be other agents or gatekeepers of information stores.
- Capacity for cooperation: A natural extension of the communication attributes is cooperation. Intelligent agents must have a collaborative spirit with other agents.
- Capacity for reasoning: The ability to perform reasoning is one of the key aspects of intelligence that distinguishes intelligent agents from other more robotic agent.
- Adaptive behavior: To maintain autonomous and reasoning capabilities, the agent must have some mechanism for assessing the current state of its external domain and incorporating this into its decision about future action.
- Trustworthiness: Essential to the acceptance of agency is a strong sense of trust that the agent can accurately represent the user, its client. This holds equally true for intelligent agents on the World Wide Web.

Based on the above attributes, we can define an intelligent agent as software which learns, infers, and cooperates if necessary with other agents or systems to solve given problems actively, autonomously and distinctively (Choi, Kim, Park, Kim, Joo, Shon, 2000). This type of intelligent agent is studied in various fields such as General Agent Theory which focuses on definition and characteristics of agent, Agent Architecture focusing on component factors, control and communication protocols of agent, and other agent application fields aiming at developing application software of practical usage.

Most research regarding job-shop scheduling deals with minimizing production cost. However, our focus here is on maximizing profit through selecting an optimal order set among those orders that were accepted under the limited production capacity. While existing scheduling studies center on the production of confirmed orders, this study considers production constraints prior to confirmation of order and tries to find ways to determine starting time for each order, to satisfy due date, and to search for a set of

orders that maximizes profit.

3. Optimazation Model

Our model for optimal order set of selection agent was formulated based on Manne's (1960) mixed-integer linear programming. In this formulation, the objective function maximizes profit while constraints are imposed to satisfy due date requirements.
The scheduling process proceeds as follows. After setting up an initial scheduling for orders requested on a certain day, a manufacturer produces all those orders if it is determined that she can meet due dates for all the orders. And if she determines that she cannot produce all the orders, production will be carried out in a sequence based on the selection process in the manner that generates most profit. In short, first stage focuses on the optimal order set that can meet due dates. In the second stage, a manufacturer plans a scheduling for those selected orders in order to minimize makespan.

3.1 Stage 1: Selecting an Optimal Order Set

The first stage selection problem makes the following assumptions:

- Setting time for a machine will be included in the processing time of some operation on the machine;
- Only one operation can be performed with one machine and an operation cannot be performed simultaneously with multiple machines; and
- When there are alternative machines, each of those machines is considered as independent machine.

A summary of symbols that are used in our model follows:

Constants

- p_i: profit generated when order *I* is produced.
- p_{ijk}: the processing time of the *j*th operation of order *i* on machine *k*.
- r_{ijk}: = 1 if the *j*th operation of order *i* requires machine *k*; = 0 otherwise.
- d_i: due date of order *i*.
- M: an arbitrary large integer.

Variables

- T_{ijk}: ≥ 0, the starting time of the *j*th operation of order *i* on machine *k.*
- O_i: =1 if order *i* is selected; = 0 if not.
- F_{max}: indicates finishing time of last job
- $Y_{(ij)(i'j')k}$: = 1 if the *j*th operation of order *i* proceeds the j′th operation of order i′ on machine k; = 0 otherwise.

The first stage to determine an optimal order set proceeds as follows.

Objective Function

MAX $\sum_i p_i O_i$

Variable Oi represents the selection of orders. If an order is selected, it has a value of 1 and if not, it has a value of 0. The objective is to maximize profits (revenues) from the selected set of orders, subject to following constraints.

Constraints

1) Operation *j* must precede operation *j+1* for order *i*.

$$\sum_k r_{ijk}(T_{ijk} + p_{ijk}) <= \sum_k r_{i,j+1,k} T_{i,j+1,k} + (1 - O_i)M \text{ ----------①}$$

2) The last operation of each order must be completed before the due date. Index *m* indicates the final operation of each order.

$$\sum_k r_{imk}(T_{imk}+p_{imk}) <= d_i \text{ ----------②}$$

3) For example, assume that there are two operations with machine *k*. If operation *j* of order *i* must precede operation *j′* of order *i′*, then it has to satisfy $T_{i'j'k} - T_{ijk} \geq p_{ijk}$. On the other hand, if operation *j′* of order *i′* must precede operation *j* of order *i*, it has to satisfy $T_{ijk} - T_{i'j'k} \geq p_{i'j'k}$. This restriction is called disjunctive constraints and since this type of constraint cannot be solved with general integer programming, an indication variable such as $Y_{iji'j'k}$ is needed (Morton, Pentico 1993). This kind of disjunctive constraint can be divided into two independent constraints using the indication variable and the traditional Big-M as follows:

$$(1-O_i)M +(1-O_{i'})M+ (M+p_{i'j'k})Y_{iji'j'k}+(T_{ijk}-T_{i'j'k}) >= p_{i'j'k} \text{ ----------③}$$

$$(1-O_i)M +(1-O_{i'})M +(M+p_{ijk})(1-Y_{iji'j'k})+(T_{i'j'k}-T_{ijk}) >= p_{ijk} \text{ ----------④}$$

3.2 Stage 2: Scheduling to Minimize Finishing Time

At the second stage, scheduling is established for the selected set of orders. The goal of the objective function is to minimize completion time for each order. The model derives from Bowman (1959) , Manne (1960), and Wagner (1959). Its description below will follow Manne's model.

Objective Function

MIN Fmax

Constraints

Completion time, instead of due dates employed in the first stage, is used as a constraint in the second stage.

$$\sum_k r_{imk}(T_{imk}+p_{imk}) <= F \text{ ----------②}$$

Table 1 summarizes objective functions and constraints of the selection agent.

[Table 1] Stage by Stage Approach of Selection Agent

Stage 1	
Objective Function	MAX $\sum_i p_i O_i$
Constraints	$\sum_k r_{ijk}(T_{ijk} + p_{ijk}) <= \sum_k r_{i,j+1,k} T_{i,j+1,k} + (1 - O_i)M$
	$\sum_k r_{imk}(T_{imk}+p_{imk}) <= d_i$
	$(1-O_i)M +(1-O_{i'})M+ (M+p_{i'j'k})Y_{iji'j'k}+(T_{ijk}-T_{i'j'k}) >= p_{i'j'k}$ $(1-O_i)M +(1-O_{i'})M +(M+p_{ijk})(1-Y_{iji'j'k})+(T_{i'j'k}-T_{ijk}) >= p_{ijk}$
Stage 2	
Objective Function	MIN F_{max}
Constraints	$\sum_k r_{ijk}(T_{ijk} + p_{ijk}) <= \sum_k r_{i,j+1,k} T_{i,j+1,k}$
	$\sum_k r_{imk}(T_{imk}+p_{imk}) <= F_{max}$
	$(M+p_{i'j'k})Y_{iji'j'k}+(T_{ijk}-T_{i'j'k}) >= p_{i'j'k}$ $(M+p_{ijk})(1-Y_{iji'j'k})+(T_{i'j'k}-T_{ijk}) >= p_{ijk}$

3.3 Experiment Results

To test the above formulation, we used the Muth and Thompson's MT(6)(1963) which is well recognized as a benchmark problem in job-shop scheduling domain. The result of our experiment was 55, which is equal to the optimal solution of MT(6). Table 2 summarizes the data and result of experiment.

[Table 2] The Result of Experiment

MT(6) Problem, Optimal Value=55												
	Machine	Man-hours	Machine	Man-hours	Machine	Man-hours	Machine	Man-hours	Machine	Man-hours	Machine	Man-hours
Job1	2	1	0	3	1	6	3	7	5	3	4	6
Job2	1	8	2	5	4	10	5	10	0	10	3	3
Job3	2	5	3	4	5	8	0	9	1	1	4	7
Job4	1	5	0	5	2	5	3	3	4	8	5	9
Job5	2	9	1	3	4	5	5	4	0	3	3	1
Job6	1	3	3	3	5	9	0	10	4	4	2	1
The Results of Experiment												
	Start Time	Finish Time	Start Time	Finish Time	Start Time	Finish Time	Start Time	Finish Time	Start Time	Finish Time	Start Time	Finish Time
Job1	5	6	10	13	30	36	36	43	43	46	49	55
Job2	0	8	8	13	15	25	28	38	40	50	50	54
Job3	0	5	5	9	9	17	18	27	41	42	42	49
Job4	8	13	13	18	22	27	27	30	30	38	46	55
Job5	13	22	22	25	25	30	39	43	51	54	54	55
Job6	13	16	16	19	19	28	28	38	38	42	54	55

Then our formulation was run to test another problem of selecting an optimal order set that maximizes profit. Its result was successful and satisfactory. The ILOG package was good up to the size of an 8 × 8 problem. But the ILOG package could not find a solution in larger problems. From these results, we concluded that a more general and robust method was needed to solve the problem which is known as NP-hard.

4. Genetic Algorithm Module

Most scheduling problems are NP-hard from the computational viewpoint. Scheduling problems in the real world tend to be solved using a combination of search and heuristics to get optimal or near-optimal solutions. Among various search methods used

for scheduling problems, the GA (Genetic Algorithm), inspired by the process of Darwinian evolution, has been recognized as a general search strategy and optimization method which is often useful in attacking combinatorial problems. The GA utilizes a population of solution in its search, giving it more resistance to premature convergence on local minima. For this reason, GA has been recognized as a proper tool for solving scheduling problems. In this section, we describe a GA approach to the JSSP (Job Shop Scheduling Problem).

4.1 Genetic Algorithm Design

We introduce the major components - representation, initialization, crossover, mutation, and replacement - of the GA which are already proven successfully (Park, 1999).

- Representation: A very important issue in building a GA for the job shop problem is to select an appropriate representation of solutions. We have adopted an Operation-Based Representation (Gen, Cheng, 1997), which is capable of coping with additional constraints of the JSSP. For an n-job m-machine, an Operation-Based Representation uses an unpartitioned permutation with m-repetition of job numbers. Here, each job number occurs m times in the permutation. The k-th occurrence of a job number refers to the k-th operation in the technological order of this job. In this way we avoid scheduling operations whose technological predecessors have not yet been scheduled. For example, consider chromosome of three jobs and three machines represented in Figure 2. An index number refers to the k-th occurrence of a job number. It is used to point to the corresponding operations in crossover. A permutation with repetition of job numbers merely expresses the order in which operations of jobs are scheduled.

[Figure 2] Representation of JSSP

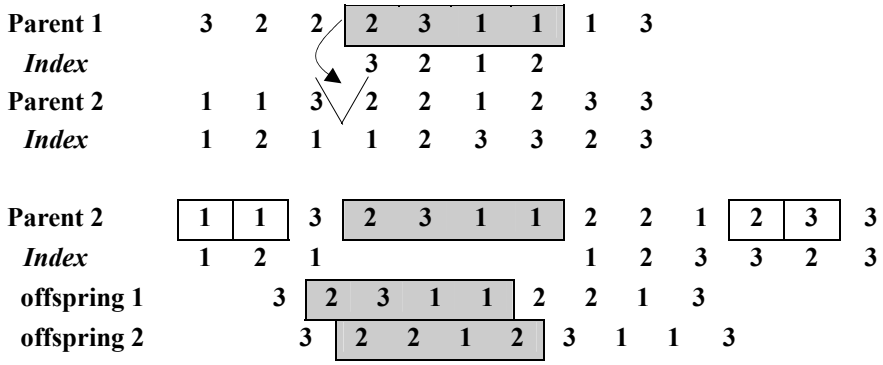
Chromosome	**[3 2 2 1 1 3 1 2 3]**
Index	**1 1 2 1 2 2 3 3 3**

- Initialization: A chromosome is made by the G&T (Giffler & Thomson) algorithm(1960) to generate active schedules. Initial population is composed of chromosomes generated by G&T.
- Selection: A tournament selection (Brindle, 1981) is adopted as selection

procedure. And a makespan is used as objective function to evaluate a chromosome.

- Crossover: The crossover operator is the most important search operator in GA. Among the various permutation operators proposed, we use MOX (Modified Order Crossover) which is modified from the techniques of GOX (Bierwirth, 1995). MOX assembles one offspring from two parental chromosomes. In MOX, as we can see in Figure 3, a substring is chosen randomly from the parent 1. MOX implants the substring into the parent 2 at the position where the first gene of substring is located. Then all genes of the substring are deleted with respect to their index of occurrence in the receiving chromosome. The process of crossover follows the same procedure after exchanging the parents 1 and 2 with the same interval. Then two offspring are evaluated and the best one is chosen.

[Figure 3] Modified Order Crossover



Changing the parents 1 and 2 with same interval generates offspring 2.

- Mutation: Mutation is just used to produce small perturbations on chromosomes in order to maintain the diversity of population. Neighborhood search-based mutation (Cheng, 1997) is used. For permutation representation, the neighborhood for a given chromosome can be considered as the set of chromosomes transformable from a given chromosome by exchanging the positions of 3 genes (randomly selected and nonidentical genes). The permutations of the genes together with remaining genes of the chromosome form the neighbor chromosomes shown in Figure 4. Subsequently, we evaluate all neighbor chromosomes and the best one is used as the offspring of mutation.

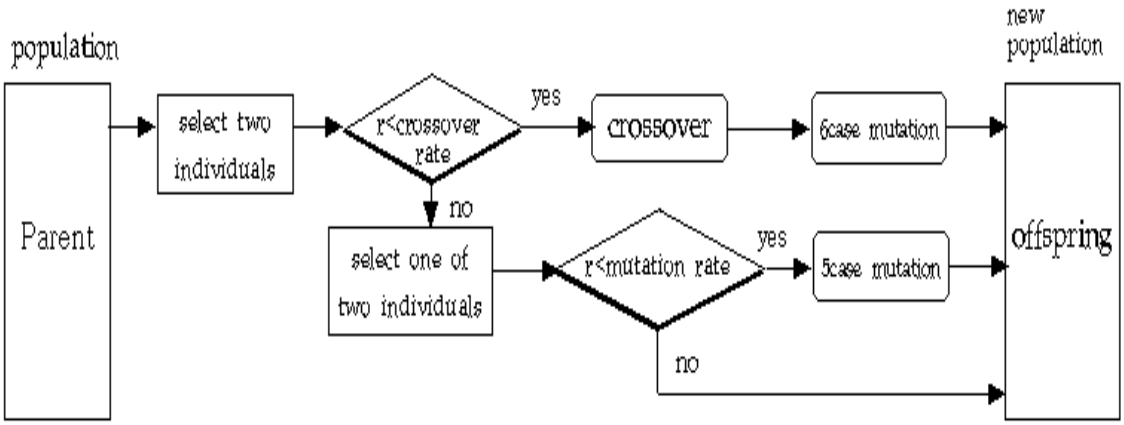
[Figure 4] Neighborhood search-based mutation

Parent chromosome	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
Neighbor chromosomes	↓				↓				↓
case 1	**1**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**3**
case 2	**2**	**2**	**3**	**1**	**1**	**3**	**1**	**2**	**3**
case 3	**2**	**2**	**3**	**1**	**3**	**3**	**1**	**2**	**1**
case 4	**3**	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**1**
case 5	**3**	**2**	**3**	**1**	**1**	**3**	**1**	**2**	**2**
case 6	**1**	**2**	**3**	**1**	**3**	**3**	**1**	**2**	**2**

In Figure 4, the mutation comparing all six cases (case 1-6) is called 6 case mutation; the mutation comparing only five cases (case 2-6) is called 5 case mutation. The former is used right after crossover, whereas the latter is used as the general mutation operator.

- Replacement: The next generation replaces the current generation only after the new population is completely created. And we use elitism in replacement. In incorporating heuristics into initialization that generates well-adapted initial population, elitism can guarantee to do no worse than the conventional heuristic does. The procedure of replacement is shown in Figure 5.

[Figure 5] The progress of replacement



4.2 Computational Results

Genetic parameters are determined through various experiments. In our experiment, crossover and mutation rates, population size, generation number, elitism size and selection probability were 0.8, 0.1, 200, 1000, 10 and 0.75, respectively.
As an example, we have used the case of Jesan Precision which is an injection-molding

company. On a certain day, Jesan Precision received three orders for an elbow, a picnic case, and a cake box as shown in Table 3.

[Table 3] The order data of Jesan Precision

Order	Due Date	Profit
Elbow	120 time	$100
Picnic case	120 time	$120
Cake box	120 time	$130

In this case, IP module could not schedule three orders simultaneously because the size of scheduling problem was too big (23 jobs, 13 machines). Thus, we scheduled three orders using GA module. When GA module schedules new orders, the GA produces completion time of each order by incremental scheduling based on the information of the previous production schedule. The completion time of three orders was 157, 146 and 156, respectively, so we cannot meet the due dates for all three orders.

[Table 4] The results of the example

Order Selection	Completion time	Due date	Observance of due date	Profit	Selection
Elbow	87	120	○	$100	
Picnic case	65	120	○	$120	
Cake Box	69	120	○	$130	
Elbow Picnic case	104 103	120 120	○ ○	$220	
Picnic case Cake Box	103 105	120 120	○ ○	$250	○
Elbow Cake Box	109 107	120 120	○ ○	$230	
Elbow Picnic case Cake Box	157 146 156	120 120 120	× × ×	$0	

The GA module searches all combinations of orders that can be completed before due dates. First, in case that three orders were produced one by one, the completion time of three orders were 87, 65 and 69 time, respectively as shown in Table 4. The completion time is equal to the results obtained by IP which was mentioned earlier. Next, in case that only two orders were produced simultaneously, the completion time of two orders were 104 (for elbow and picnic case), 105 (for picnic case and cake box), and 109 time

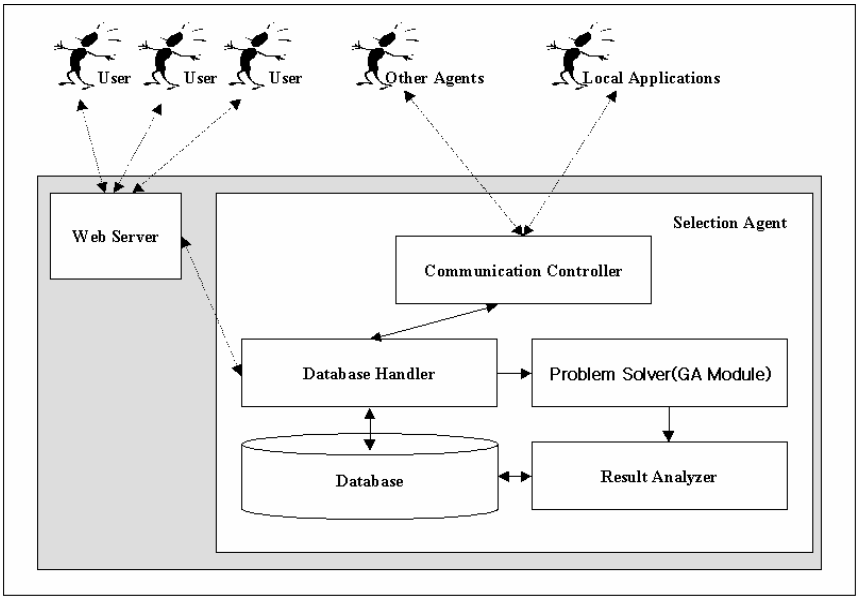
(for elbow and cake box), respectively. As a result, we are able to find the product capability of two orders, and decide to produce picnic case and cake box with the bigger profit.

Here, we can confirm that GA is more suitable as a scheduling module of selection agent

5. Architecture of Selection Agent and Implementation

Figure 6 below shows the overall architecture of selection agent.

[Figure 6] Architecture of Selection Agent



- Communication Controller: Communication Controller is a module that controls communications to and from other agents or inside applications of production related agents, consisting of message converter, message queuing, and message manager. Functions of each module are as follows.
  - Message converter: Every message is transmitted through the message converter and the message is changed to TCP/IP protocols. It also controls the connection with other agents.
  - Message queuing: It manages outbound and inbound messages permitting only valid messages to pass.
  - Message manager: It inspects message forms based on all the hierarchies of KQML (Knowledge Query and Manipulation Language)(Finin, Mckay,

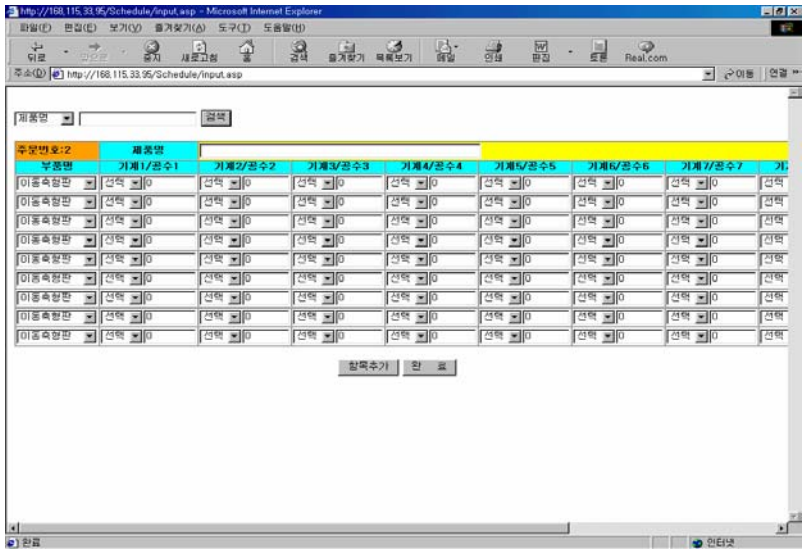
Fritzon, 1992), and initializes the reasoning engine to generate a response message according to the query message.

- Problem Solver: Problem Solver is a selection agent engine that executes a GA program.
- Result Analyzer: The result analyzer saves results, which are the outcome of Problem Solver (GA module), into database and provides users with graphic views. Also, users can manually modify the results from this module whenever necessary.
- Database Handler: The database handler saves and manages input data and result data.

The procedure of application service for small and medium manufacturers provided by selection agent through the Internet is as follows.

1) Users connect to web server, and then input the data for orders, and request desired services. Figure 7 below shows a screen that inputs data.

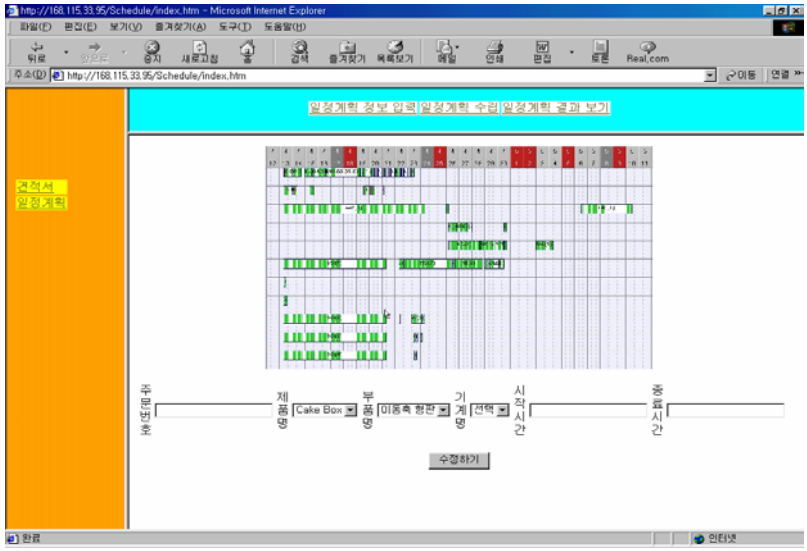
[Figure 7] A Screen of Data Input



2) Upon completion of inputting data and requesting services for selection agent, the problem solver of selection agent begins its execution.
3) Once problem solver executes, the result is transmitted to the result analyzer and

the result analyzer provides users with the result of scheduling. Figure 8 below shows a screen with the result of scheduling.

[Figure 8] A Screen of Gantt Chart



We implement selection agent by single agent for selecting optimal order set which has the abilities of autonomy, intelligence, and adaptation for dynamic environment. But if there is other agent like as buyer agent, selection agent could communicate with buyer agent complaint with KQML. 3 orders are received from buyer1, buyer2, and buyer3. The KQML message is as follows.

```
(ask-one
        :sender         buyer1
        :receiver       seller
        :language       KQML
        :reply-with     order1
        :contents       (product_name       picnic_case,
                         due_date           2001-06-25,
                         price              $30000))

(ask-one
        :sender         buyer2
        :receiver       seller
```

```
        :language       KQML
        :reply-with     order2
        :contents       (product_name       cake_box,
                        due_date            2001-06-25,
                        price               $40000))

    (ask-one
        :sender         buyer3
        :receiver       seller
        :language       KQML
        :reply-with     order3
        :contents       (product_name       elbow,
                        due_date            2001-06-25,
                        price               $35000))
```

When manufacturer couldn't produce all of them. Selection agent should send the message whether produce or not as follows.

```
    (tell
        :sender         seller
        :receiver       buyer1
        :language       KQML
        :reply-with     order1
        :contents       (product_name       picnic_case,
                        due_date            2001-06-25,
                        price               $30000,
                        contract_confirm    NO))

    (tell
        :sender         seller
        :receiver       buyer2
        :language       KQML
        :reply-with     order2
        :contents       (product_name       cake_box,
                        due_date            2001-06-25,
                        price               $40000,
```

```
                    contract_confirm    YES))

(tell
        :sender         seller
        :receiver       buyer3
        :language       KQML
        :reply-with     order3
        :contents       (product_name       elbow,
                        due_date            2001-06-25,
                        price               $35000,
                        contract_confirm    YES))
```

6. Conclusion

Most of small and medium manufacturers in Korea cannot perform production management due to their small scale and inadequate equipment. In order to solve these problems and to increase sales power of small and medium companies in the age of the Internet, we have proposed an architecture and methodology of the selection agent. At first, we have developed an integer programming formula to acquire an optimal set of orders which maximizes profit and to schedule production processes which minimize production cost. Like other scheduling algorithms, it is an NP-hard problem. We couldn't solve the real world problem using IP model. To cope with the complexity of scheduling problem as well as to reflect dynamic environment, we have adopted the Genetic Algorithm, and it results in satisfactory solution.

Although we have implemented the main engine of sales agent proposed using GA, we didn't reflect the feature of agent. Since it is an ongoing research, it will be able to be accomplished in the near future.

References

Choi, J. M. "Overview and Research Direction of Agent", *Korea Information Science Society Review* 15, No. 3, 1997, pp. 7-15.

Bierwirth, C., "A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms", *OR-Spektrum, Special Issue: Applied Local Search, Pesch, E. and Vo, S.(eds)*, 17(213), 1995, pp. 87-92.

Bowman, E. H., “The Schedule-Sequencing Problem”, *Operations Research 7*, No. 5, 1959.

Brindle, A., Genetic Algorithms for Function Optimization, Unpublished Ph. D. thesis, University of Alberta, Edmonton, Canada, 1981.

Cheng R., “A study on genetic algorithms-based optimal scheduling techniques”, Ph.D. thesis, Tokyo Institute of Technology, 1997.

Choi, H. R., Kim, H. S., Park, Y. J., Kim, K. H., Joo, M. H., and Shon, H. S., “A Sales Agent for Part Manufacturers: VMSA”, *Decision Support Systems* 28, 2000, pp. 333-346.

Finin, T., Mckay, D., and Fritzson, R., the KQML Advisory Group(Eds.), “An overview of KQML: A Knowledge Query and Manipulation Language”, Mar., 1992.

Franklin, S., and Graesser, A., “Is it an Agent or just a program?: A Taxonomy for Autonomous Agents”, *Proceedings of the 3rd International Workshop on Agent Theories, Architecture, and Language*, 1996.

Gen, M. and Cheng, R., *Genetic algorithms and engineering design*, New York, John Wiley & Sons, 1997, pp. 203-204.

Giffler, J. and Thompson, G.L., “Algorithms for Solving Production Scheduling Problems”, *Operations Research* 8, 1960, pp. 487-503.

Manne, A. S., “On the Job-Shop Scheduling Problem”, *Operations Research 8*, No.2, 1960.

Morton, T. E., and Pentico, D. W*., Heuristic Scheduling System*, Jhon Wiley & Sons, pp. 366-369, 1993.

Muth, J. F. and Thompson, G. L., Industrial Scheduling, Prentice-Hall, Englewood Cliffs, N.J, 1963

Nissen, M., Intelligent Agent: A Technology and Business Application Analysis", 1995.

Park, B. J., "A Development of Hybrid Genetic Algorithms for Scheduling of Static and Dynamic Job Shop", Ph.D. thesis, Department of Industrial Engineering, University of Dong-A. 1999.

Wagner, H. M., "An Integer Linear-Programming Model for Machine Scheduling", *Nav. Res. Log. Quart. 6*, No. 2, 1959.