A Hierarchical System for Dynamically Solving Planning and Scheduling Problem in a Flexible Manufacturing System

Pei-Sen Liu and Li-Chen Fu

Department of Computer Science & Information Engineering
National Taiwan University, Taipei, Taiwan, R.O.C.

ABSTRACT

There are inevitably time-varying factors existing in a flexible manufacturing system (FMS). In this paper, a system with a two-level structure (high and low level) for dynamically solving the problem of planning and scheduling in an FMS altogether is presented. The problem is first formulated as the determination of an optimal routing assignment of p automated guided vehicles (AGV's) among m workstations to accomplish N tasks, facing several possible dynamical situations, e.g. change of due date or breakdown of some workstation(s). The hierarchical system is then built to solve this optimization problem in a dynamical manner. The low level aims to solve the routing problem for AGV's among workstaions given a set of AND/OR graphs which represent tasks to be processed. On the other hand, the high level, via a rule based system, provides necessary data for low level use and simultaneously determines principles as how to respond to occurrence of some unexpected events. It is shown that a near-optimal solution can be derived with moderate computation time, which allows operation in an FMS to be more flexible.

I. Introduction

An FMS is a large complex system consisting of many interconnected components of hardware and software [1][2]. The necessary planning and decision control of an FMS include balancing the workload, responding to changes, scheduling and dispatching, and managing tools and materials, etc. Several aspects of current FMS planning and decision control system have been discussed in [2]-[6].

In an FMS, a workstation (or so called cell) is defined as the smallest building block of the system consisting of a computer controlled machine. A basic design principle of an FMS is to have as many identical workstations as possible to enhance flexibility. Therefore, there may be more than one sequence of operations to produce a part, which implies that "planning" on the choice of a single sequence out of all possible candidate sequences has to be performed. In addition, it is the AGV that are responsible of carrying workpieces among workstations for each operation. Hence a good scheduling method will be required to determine the routing assignment for AGV's. But as has been known that the complexity of solving this problem is considerably high if its so called "optimal" solution is attempted. Thus, introduction of some good heuristics so as to reduce the complexity would be more favorable, especially in a dynamic environment such as an FMS where occurrence of unexpected events is usual.

Several approaches to planning, decision control, and scheduling of an FMS have been suggested in [7]-[12],[16]. However, the proposed works either can not solve the problem of planning and scheduling dynamically, or are not suitable for solving the routing problem of AGV's in an FMS.

This paper formulates the problem of dynamic planning and scheduling as the one of determining the optimal routing assignment of p AGV's among m workstations to accomplish N tasks in an FMS. The approach used here is a hierarchical and dynamic one which solves the problems of planning and scheduling as a whole. Especially it can handle some occasional situations such as breakdown of some workstation(s), as well as can respond to some changes, for example, of production target.

The paper is organized as follows: in section II, we formulate the problem and describe the optimality criterion for the solution; in section III, we briefly introduce the structure of our hierarchical planning and scheduling system, and describe the necessary information for the system; in section IV, we explain how the low level subsystem works and the techniques used in this level, including A^* search algorithm and minimax strategy; in section V, we describe the function of the high level subsystem and its most important rule based system; in section VI, some computational results are shown to illustrate the performance of this hierarchical system.

II. Problem Formulation and Optimality Criterion

The main problem in an FMS is usually divided into two parts: planning, or process routing, which is the selection of a sequence of operations; and scheduling, which is the assignment of time and resources. Here we define our problem as the determination of an optimal routing assignment of p AGV's among m workstations to accomplish N tasks in order to minimize the total completion time.

Suppose there are k kinds of the total N tasks and the number of the *i*th kind is N_i, $1 \le i \le k$, so that $N = N_1 + N_2 + \ldots + N_k$. Each task is to let an AGV carry necessary materials (parts or subparts) and to follow the assigned route among workstations so that every subtask of the task can be successfully processed by the selected workstations. Thus, to complete N tasks the routes for total p AGV's among m workstations have to be assigned. Moreover, the total completion time of a task is the sum of total execution time (through workstations), total travelling time of an AGV (among workstations), and total waiting time (when two or more AGV's arrive at the same workstation). Now due to the fact that each task will require a sequence of processing by some selected workstations, performing the routing assignment is equivalent to solving the

CH2809-2/89/0000-0077 $1.00 © 1989 IEEE

problem of planning. Later in the sequel, how to minimize the total completion time will be shown to be our main consideration for assigning routes for those AGV's. This indicates that, from our previous definition, the planning problem can also be treated as a scheduling problem. Consequently, we can define the problem of planning and scheduling together in an FMS as the *routing assignment* problem. The following are necessary assumptions.

Assumptions :

1) All AGV's are the same. Every AGV can carry all kinds of materials necessary for all kinds of tasks.

2) All AGV's are travelling at constant known speed.

3) The travelling time for each AGV between workstation W_i, W_j, and W_k satisfies the triangle inequality.

4) All workstations along with all paths connecting them form a complete graph[18]. Each path can accommodate only one AGV at a time.

5) There are no precedence constraints between any two tasks, but every task can be decomposed into some related subtasks. The general AND/OR graph [13][14] is used to represent these subtasks [18].

6) The number of every part to be produced and the due date for every kind of part are given at any time, but may subject to variation as the manufacturing proceeds.

7) Tasks are nonpreemptive.

The design of a mathematical model for the routing assignment problem follows the steps shown below.

A. Representation of the System Model

A material handling system of an FMS can be represented by a triplet $S = (A, W, T)$, in which the arguments represent the set of *AGV's* , the set of *workstations*, and the set of *task* graphs respectively. Specifically, $A = \{v_1, v_2, ..., v_p\}$, where v_i denotes the *i*th AGV, and all p AGV's are the same except that they may run at different speed; $W = \{W_1, ..., W_q, W_{q+1}, ..., W_m\}$, where W_i, $1 \le i \le q$, denote the real workstations and W_j, $q+1 \le j \le m$, denote the terminals for AGV's (loading and unloading places); and $T = \{T_1, T_2, ..., T_k, N_1, N_2, ..., N_k, D_1, D_2, ..., D_k\}$, where T_i, N_i, and D_i denote respectively the AND/OR task graph, the number of task, and the due date of the *i*th kind of task, $1 \le i \le k$, and $N = N_1 + ... + N_k$.

Remarks:

(1) Later in the sequel, we will not distinguish the real workstations from terminals, and simply call W_i the *i*th workstation, $1 \le i \le m$.

(2) We further denote $T_i = (V_i, E_i)$, the *i*th task graph, where V_i represent the set of vertices and E_i the set of directed links.

Let WTT^r be the symmetric matrix to represent the travelling time between any two workstations for AGV v_r, where the entry $WTT^r_{i,j}$ denotes the travelling time spent by that AGV when it travels from workstation i to workstation j, $1 \le i,j \le m$. Every task graph $T_i = (V_i, E_i)$ is an general AND/OR graph with directed edges. The directed edge from vertex i to vertex j indicates that the subtask to be processed by workstation i must go before the subtask to be processed by workstation j. An OR branches indicates that only one of the subtasks which immediately follow the branch needs to be done. An AND branches indicates that all subtasks following the branch should be completed but the order is not important (since there are no precedence or dependency relationship among them).

B. Cost Function

The cost function for a particular task when assigned to certain AGV is the sum of the aforementioned execution time, travelling time, and waiting time spent by that AGV. Now suppose that the AGV v_i is assigned to perform tasks $t^i_1, ..., t^i_{p_i}$ sequentially, then its travelling sequence is denoted $O^i = (O^i_1, O^i_2, ..., O^i_{n_i})$ where $O^i_j \in W$, $1 \le j \le n_i$, and n_i is the total number of workstations travelled. Given this notation, we are now ready to define travelling time, execution time, and waiting time throughout a task execution by an AGV, say, v_i in detail as follows.

Definition 1: Let TT^i_j denote the time spent by the AGV v_i when travelling from workstation O^i_j to workstation O^i_{j+1} out of the sequence O^i, $1 \le j \le (n_i - 1)$. Then the travelling time of v_i to accomplish the sequence of tasks $t^i_1, ..., t^i_{p_i}$ is defined as:

$$TT^i = \sum_{j=1}^{n_i - 1} TT^i_j$$

Definition 2: Let E^i_j denote the time required by workstation O^i_j to complete its processing, $1 \le j \le n_i$. Then the task execution time corresponding to the travelling sequence O^i is defined as:

$$ET^i = \sum_{j=1}^{n_i} ET^i_j$$

Definition 3: If AGV v_i and AGV v_j arrive at a workstation at the same time or a workstation is doing some subtask for AGV v_j when AGV v_i arrives, then either of the two AGV's, v_i and v_j, has to wait or v_i has to wait till the workstation finishes the current subtask. Hence we let WT^i_k be the waiting time of the AGV v_i at the workstation O^i_k *from the travelling sequence* O^i so that the total waiting time of the AGV v_i to complete the travelling sequence O^i is defined as:

$$WT^i = \sum_{i=1}^{n_i} WT^i_j$$

As a result of the above definitions, the total completion time $T_i(S)$ spent by the AGV v_i corresponding to the routing assignments S for all AGV's is the sum of travelling time, task execution time, and waiting time as follows:

$$T_i(S) = TT^i + ET^i + WT^i$$

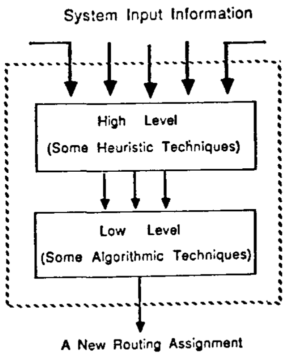


Figure 1. The Structure of the Dynamic Hiererchical Planning and Scheduling System.

TABLE 1.

TRAVELLING TIME MATRIX

	W1	W2	W3	W4	W5	W6	W7
W1	0	2	3	2	4	3	2
W2	2	0	2	3	3	3	3
W3	3	2	0	3	3	2	3
W4	2	3	3	0	4	3	3
W5	4	3	3	4	0	3	4
W6	3	3	2	3	3	0	3
W7	2	3	3	3	4	3	0

for AGV v_1

	W1	W2	W3	W4	W5	W6	W7
W1	0	2	3	2	3	3	2
W2	2	0	2	2	3	3	3
W3	3	2	0	3	2	2	3
W4	2	2	3	0	3	3	3
W5	3	3	2	3	0	2	3
W6	3	3	2	3	2	0	2
W7	2	3	3	3	3	2	0

for AGV v_2

C. Optimality Criterion

Let Ω denote all possible routing assignments for p AGV's to accomplish all tasks before due dates. Then our optimality criterion *is to find an optimal routing assignment S such that:*

$$S = \underset{S' \in \Omega}{\operatorname{argmin}} \{ \max_{1 \leq i \leq p} (T_i(S')) \}$$

It is well known that this problem is an NP-complete problem. Moreover, the environment of a manufacturing system may be changing dynamically. Therefore, it may be impractical to spend a lot of time to find an optimal assignment in advance of real running. Therefore, we will use a dynamic method to find a "near optimal" solution.

III. A Hierarchical Planning and Scheduling System

To dynamically solve our problem, in this section, we propose a hierarchical system whose structure and environment is shown in Figure 1. More detailed explanations are given in the following.

A. System Input :

The information that should be input to the hierarchical system can be classified into the following two groups:

(1) *Initial Information* :

This describes the initial state of the environment of an FMS, but it may be changed afterwards. It usually includes the following:

(i) position of each workstation;

(ii) distance between any two workstations;

(iii) number of total AGV's;

(iv) travelling speed of each AGV;

(v) current number of products of each kind to be manufactured;

(vi) *manufacturing process and due date of each kind of product;*

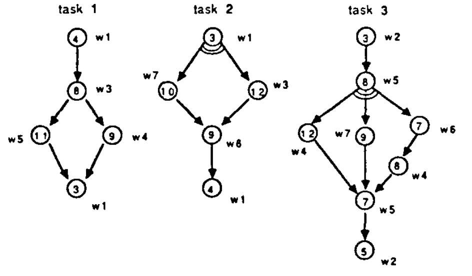


Figure 2. AND/OR Task Graphs.

	task 1	task 2	task 3
Number	100	50	75
Due Date	5700	5350	4000

Table 2. Task Information.

For illustration, we give an example of such input information in Table 1,2 and Figure 2, where the former indicates the necessary information about workstations and AGV's through travelling time matrices whereas the latter describes those about each kind of products. In this example, the FMS includes two AGV's, seven workstations, and three different kinds of tasks. In particular, the numerics in each node of graphs in Figure 2 denotes the amount of execution time needed in the corresponding workstation.

(2) *Occasional Events* :

These events may occur occasionally but unexpectedly due to the dynamic nature of an FMS. They include the following:

(i) change of the condition of some AGV('s);

(ii) breakdown of some workstation(s);

(iii) change of position of some workstation(s);

(iv) change of due date(s) of some product(s);

(v) addition of another kind of product to be manufactured;

B. System Structure

The hierarchical system is a two-level structure, namely, a high level subsystem on the top of the low level one. Their respective functions and solution techniques are described as follows.

(1) *High Level* :

This level can be regarded as the decision supporting part of the overall system. Two kinds of problems to be solved here are the following: (1) Given an AGV which is freed at a particular instant of time (either because it has just carried out a task or because it is newly added in), which unprocessed task should be assigned to it ? (2) When some occasional event(s) suddenly occur, what kind of correction to the already determined routing assignment should be enforced ? Since this level involves the problems with policies and with unpredictable events, it is seen to be more efficient to use some heuristic techniques to solve them. Therefore, a rule based system using forward chaining control is adopted here as the main solution technique of this subsystem.

(2) *Low Level* :

The problem for this level can be precisely phrased as follows: When each AGV is assigned to one of the tasks (several could be of the same kind) to be processed (due to the decision made in the high level) and given the most current state information of the environment, what are the optimal routes for all the AGV's so that the previously mentioned optimality criterion is to be or close to be achieved. Clearly, this problem is fairly a deterministic one and, hence, algorithmic techniques are preferable for solving it. This, then, leads to our choice of A* search algorithm and minimax strategy as the solution technique in this level.

IV. Low Level Algorithmic Method

The structure of this level is shown in the diagram in Figure 3. All the data are stored in the data base which includes the following: 1) the current position and the condition (free or busy) of each available AGV, 2) the state of every workstation, 3) the kind of task to which each AGV is assigned, and 4) the AND/OR graph of every task. These data are frequently updated through the high level instructions to reflect the current environment. With the most current information stored, the low level then solves the routing problem via an algorithmic method.

Before the routing assignment problem can be solved, it is more convenient to transform it into a state space search problem so that the powerful A^* algorithm[14] and minimax strategy[15] can be used. In this level, the problem to be treated here is only a part of the whole original problem. The solution in this case is simply an optimal routing assignment for p AGV's among m workstations to accomplish p tasks (in contrast with the original N tasks). The state space search method for this level and the heuristic estimation procedure are stated in [17][18].

After computing the heuristic cost of every node in the task graph for all AGV's, we can then calculate every cost estimate $f_i(n)$ for every AGV v_i at run time. Finally, since the evaluation function f(n) defined before satisfies the minimax strategy, i.e.:

$$f(n) = \max_{1 \le i \le p} f_i(n)$$

our principle for expanding nodes in A^* search algorithm will obtain an optimal solution. Now we have shown how to solve the fixed problem by using A^* search algorithm and minimax strategy in the low level of our system.

By assumption 4), every passage connecting one workstation to another can allow AGV's running only in one direction at a time. If an AGV is already on that passage and is heading toward a workstation while the other AGV is ready to leave that workstation by that passage but with converse direction, the the waiting time for the departing AGV should be accounted for in the calculation of the evaluation function. But when more than two AGV are running on the same passage and the one in the rear is running at a faster speed than that of the one in the front, similar waiting time should also be considered.

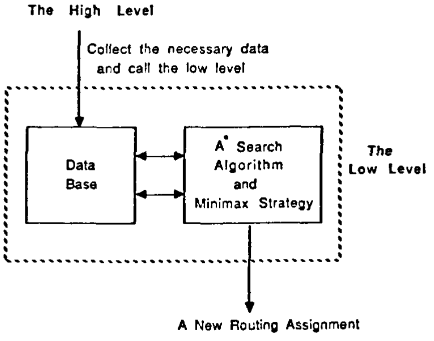


Figure 3. The Structure of the Low Level.

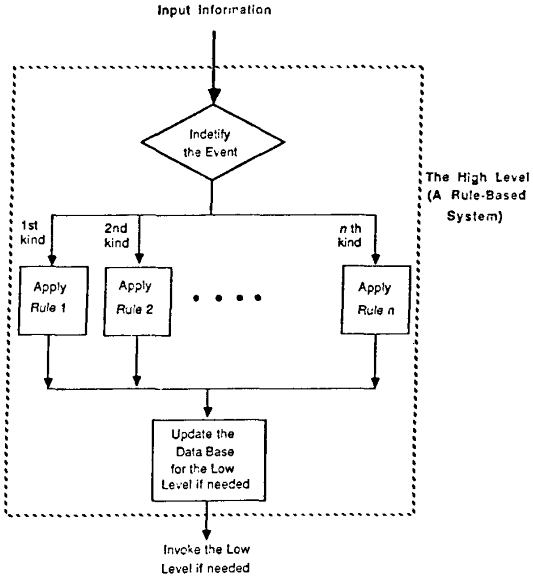


Figure 4. The Structure of the High Level.

V. High Level Heuristic Method

This level is responsible for making decisions and sending useful data to the low level subsystem. Figure 4 shows the structure of this level whose process flow can be summarized into the following steps:

Step 1. Identify the occasional event(s) which just occur(s);

Step 2. Change the information about the entire environment;

Step 3. Take some proper measures to handle the event;

Step 4. Update the data base of the low level subsystem;

Step 5. Invoke the low level if necessary to obtain a new routing assignment.

A. Basic Method

Whenever some occasional event occurs, including that where some task has been just completed, the *optimality principle* will require that all the subtasks which are in process or ready to be processed be taken into reconsideration. This means that if some AGV, say, v_i is assigned a task with its corresponding AND/OR graph containing m_i nodes, V_1, ..., V_{m_i}, and currently this AGV is performing the subtask corresponding to the node V_{i_y} ($1< i_y< m_i$), then we must consider all the following subtasks corresponding to nodes $V_{i_{y+1}}$,..., V_{m_i} together in order to find a better routing assignment. Hence, the predetermined route for AGV v_i may be changed after a new invoke of the routing assingment process of the low level. A detailed procedure used to update the data base, especially the state description, of the low level subsystem is given in [18].

B. Rule Based System

One of the major functions of the high level subsystem is to handle occasional events and make responses. Since these events vary drastically, their handling processes are considerably different. Therefore, it is especially appropriate to use a rule based system (

```
( P rule1
   ( acknowledge the change of due date of some kind of product)
 -->
   ( update the due date)
   ( notify the supervisor)
)

( P rule2
   ( an AGV is freed from its proceding task)
 -->
   ( update the condition of that AGV)
   ( reduce the number of the unprocessed tasks of the same kind)
   ( assign this AGV to some unprocessed task by some heuristics)
   ( update the data base of the low level subsystem)
   ( invoke the routing assignment process of the low level)
)

( P rule3
   ( a workstation breaks down)
 -->
   ( change the initial information about the system)
   ( notify the supervisor)
   ( send warning signal)
   ( update the data base of the low level subsystem)
   ( invoke the routing assignment process of the low level)
)
   .
   .
```

Figure 5. Examples of Production Rules

consisting of production rules) as shown in Figure 4 to deal with this situation. The production rule is, in general, of the following form:

IF < cond.1> ... < cond.n> THEN < act.1> ... < act.m>

For illustration, examples of the production rules are given in Figure 5. Among them, the first one express the subsequent action when the due date of some kind of product is changed; the second one lists a set of measures whenever an AGV is freed from its preceding task; the last one takes care of the situation where the breakdown message of any workstation is received. In the first example, no immediate invoke of a new routing assignment process is needed, but the rest of two examples will require such a reaction to get a new routing assignment.

For a number of other similar situations, analogous production rules can be generated. Hence, it is this rule based system which makes the overall planning and scheduling system extremely dynamic.

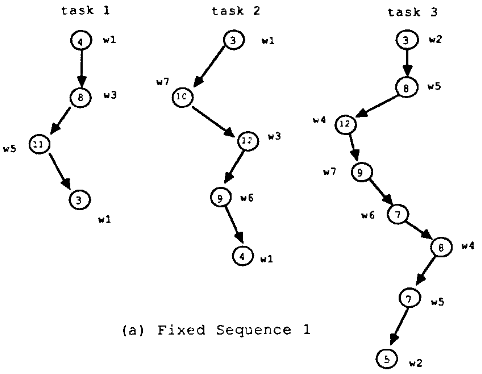
VI. Computer Simulation Examples

In this section, we implement a simulation program of the hierarchical planning and scheduling system on VAX 8530. Comparison between results of our approach and those from the other methods is performed and shown. For simplicity, we assume there is no occasional event which occurs unexpectedly so that the FMS is running in a normal situation. The necessary information about the problem is given in Table 1,2 and Figure 2, and the simulation results are shown in Table 3. In the first column of Table 3, there are four different kinds of methods and each with six different conditions (about different due dates and different heuristics) listed in the first row. The details about these methods and conditions are:

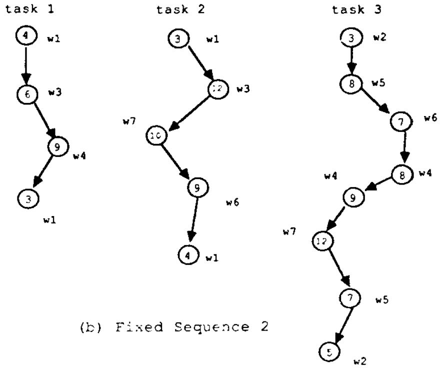
Methods:

(1) Use the hierarchical planning and scheduling system presented in this paper to solve the problem as shown in Figure 2.

(2) Use a method similar to (1) with the difference described as follows: whenever an AGV is freed, a new task is assigned to it and a new route for that AGV is established through the use of A^* search algorithm, but all predetermined routes for other busy AGV's remain unchanged.



(a) Fixed Sequence 1



(b) Fixed Sequence 2

Figure 6. Two Fixed Travelling Sequences.

(3) The method here is similar to (1) except that processing sequence to accomplish every kind of task is fixed as shown in Figure 6(a).

(4) This method is similar to (3) but the processing sequence is now changed as shown in Figure 6(b).

Conditions:

(1) Different *due dates* : Three different types of due dates for three kinds of tasks are assumed.

A *rule based system* : The rules contained here involve the due dates, the estimated completion time, and the number of every kind of tasks.

Earliest *Due Date* (*EDD*) *Principle* : The processing order of different kinds of tasks is based on the knowledge of due dates.

The measures used here are the total time spent to accomplish every kind of task and that to accomplish all these tasks. The test results show that our system with AND/OR task graphs (method 1) performs much better than the others. In addition, our rule based module is preferable to the EDD principle, since some results given by EDD principle can not even finish all the tasks before the due date as opposed to ours. Moreover, since the number of AGV's in a real FMS is usually small, and the computer run-time spent in A^* search algorithm heavily depends on the number of total AGV's and the size of every task graph, the total computational time can,

Cond. / Completion time / Method	The due date is 770 , 770 , 770 respectively		The due date is 780 , 680 , 510 respectively		The due date is 600 , 780 , 600 respectively	
	Rule-Based	EDD rule	Rule-Based	EDD rule	Rule-Based	EDD rule
Method 1	752 720 740	717 251 539	712 600 489	825 649 455	560 704 589	617 809 455
Method 2	831 799 807	819 253 640	810 746 595	807 647 448	736 834 763	629 827 448
Method 3	923 888 932	971 294 813	955 821 572	967 773 589	704 941 725	761 971 597
Method 4	840 806 823	841 260 665	839 674 511	841 663 465	643 842 673	641 842 645

(a) N1=10 , N2=10 , N3=10 .

Cond. / Completion time / Method	The due date is 5300 , 5300 , 5300 respectively		The due date is 5700 , 5350 , 4000 respectively		The due date is 4500 , 5800 , 4500 respectively	
	Rule-Based	EDD rule	Rule-Based	EDD rule	Rule-Based	EDD rule
Method 1	5136 5104 5127	4984 1221 3382	4984 4764 3360	5771 4171 3045	4237 5149 4261	4719 5879 3045
Method 2	5915 5883 5892	5837 1213 4176	5837 5600 4482	5845 4147 3015	5098 5612 5141	4685 5843 3015
Method 3	7212 7205 7198	7214 1434 5214	7201 5216 3883	7226 5231 3858	5361 6795 5394	4802 6038 4832
Method 4	6090 6067 6056	6065 1260 4289	5990 5224 3580	6051 4252 3089	4802 6038 4832	4865 6066 3089

(b) N1=100 , N2=50 , N3=75 .

TABLE 3. Simulation Results.

in fact, be more satisfactory. Consequently, this method constitutes a promising planning and scheduling algorithm in a real environment of an FMS.

VII. Conclusion

In this paper, the problem of planning and scheduling altogether in an FMS is formulated as the determination of an optimal routing assignment for automated guided vehicles. Due to the dynamic nature of an FMS, a hierarchical system is built to solve this routing assignment problem in a dynamic manner. Such a system integrates the advantages of both algorithmic and heuristic techniques and, hence, can be more general and more efficient. For illustration, computer simulation examples are provided and their results shown in Table 3 are, in fact, quite satisfactory. Furthermore, since the number of AGV's in an FMS is usually very small, the total computational time is also economical. Therefore, application of this method to an FMS in a real environment will be quite promising.

References

[1] Simpson, J. A., R. J. Hocken, and J. S. Albus, "The automated manufacturing research facility of the National Bereau of Standards," *J. Manuf. Syst.*, vol. 1, pp. 17-32, 1982.

[2] Sui, R. and C. K. Whitney, "Decision supporting requirements in flexible manu-facturing," *J. manuf. Syst.*, vol. 3, no. 1, pp. 61-69, 1984.

[3] Chen, P. E. and J. Talavage, "Production decision support system for computerized manufacturing system," *J. Manuf. Syst.*, vol. 1, no. 2, pp. 157-168, 1982.

[4] Whitney, C. K., "Integration in FMS design and control," in *Proc. ASME Computer in Engineering Conf.*, pp. 355-360, 1986.

[5] Young, R. E., "Planning and control requirements for flexible manufacturing systems," in *Proc. ASME Computers in Engineering Conf.*, pp. 347-353, 1986.

[6] Ballakur, A. and H. J. Steudel, "Integration of job shop control systems: A state-of-art review," *J. Manuf. Syst.*, vol. 3, no. 1, pp. 71-88, 1984.

[7] Stecke, K. E. and T. L. Morin, "The optimality of balancing workload in certain types of flexible manufacturing system," *Europe. J. Oper. Res.*, vol. 20, pp. 66-72, 1985.

[8] Ho, Y. C. and X. R. Cao, "Performance sensitivity to routing changes in queuing networks and flexible manufacturing systems using perturbation analysis," *IEEE J. Robotics Automation*, vol. RA-1, no. 4, Dec., 1985.

[9] Bermen, O. and O. Maimon, "Cooperation among flexible manufacturing systems," *IEEE J. Robotics and Automation*, vol. RA-2, no. 2, pp. 24-30, Mar., 1986.

[10] Fox, B. R. and K. G. Kempf, "Opportunistic scheduling for robotics assembly," in *Proc. IEEE Inter. Conf. on Robotics and Automation*, pp. 880-889, 1985.

[11] Xiaodong Xia George A. Bekey, "SROMA: An adaptive scheduler for robotic assembly system," in *Proc. IEEE Inter. Conf. on Robotics and Automation*, pp. 1282-1287, 1988.

[12] Jane C. A., T. Govindaraj, and M. M. Christine, "Decision models for aiding FMS scheduling and control," *IEEE Trans. Syst. Man Cybern.*, vol. 18, no. 5, pp. 744-756, Sep./Otc., 1988.

[13] L. S. Homem de Mello and A. C. Sanderson, "AND/OR graph representation of assembly plans," in *Proc. AAAI*, pp. 1113-1119, 1986.

[14] N. J. Nilsson, *Principle of Artificial Intelligence*, Palo Alto, CA: Tioga, 1980.

[15] Shen, C. C. and W. H. Tasi, "A graph matching approach to optimal task assignment in distributed computing systems using a minimax criterion," *IEEE Trans. Comput.*, vol. C-34, pp. 197-203, Mar., 1985.

[16] Chun, C. L., C. S. George and C. D. Mcgillem, "Task assignment and load balancing of autonomous vehicles in a flexible manufacturing system," *IEEE J. Robotics and Automation* vol. RA-3, no. 6, Dec., 1987.

[17] Liu, Pei-Sen and L. C. Fu, "Planning and scheduling in a flexible manufacturing system using a dynamic routing assignment method for automated guided vehicles," in *Proc. IEEE Inter. Conf. on Robotics and Automation*, May, 1989.

[18] Liu, Pei-Sen and L. C. Fu, "A Hierarchical System for Dynamically Solving Planning and Scheduling Problem in Flexible Manufacturing System," Technical Report NUTCSIE 88-05. Department of Computer Science and Information Engineering, National Taiwan University, 1988.