1998 International Computer Symposium
Workshop on Artificial Intelligence
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

GENETIC ALGORITHM BASED DISPATCHING RULES FOR FMS SCHEDULING

Ming-Shing Hsieh[1,2], *Yong-Huai Huang*[1], *and Din-Chang Tseng*[2]

[1]Department of Information Management,
Tamsui Oxford University College, Tamsui, Taiwan, R.O.C
Email: shiehms@ms21.hinet.net

[2]Institute of Computer Science and Information Engineering
National Central University, Chung-Li, Taiwan, R.O.C

ABSTRACT

In this paper, we propose a genetic search algorithm based on the timed Petri net model in order to find an optimal or near optimal dispatching rules under specific performance measures and restrictions. The chromosomes that start with an initial marking represent all conflict resolutions for resources assignment and resources dispatching order. The fitness associated with penalties of each trial solution is assessed by the measure of fitness function. Our experiments show that the proposed approach does represent a good alternative for dispatching strategies of FMS problems.

1. INTRODUCTION

A production system usually contains multiple concurrent flows of job processes and often exploits shared resources. In general, Due to the NP-complete nature of the scheduling problem of an flexible manufacturing system (FMS) [1], generating a dispatching schedule to efficiently allocate resources for manufacturing of products is a complex and difficult task. Owing to the sophisticated process routing and fairly complicated constraints, the simulators for FMS are so expensive with long computing time that they can hardly be taken into consideration for practical applications.

Petri nets are recognized to be successful tools for modeling FMS because of the advantages such as the graphical nature that is to embody the static structure and the dynamic behavior, and the availability of the mathematical analysis techniques [2]. Methods that incorporate Petri nets to solve the scheduling problems in an FMS start with Petri nets modeling and then employ a heuristic to find the optimal solutions. However, it is a complex and difficult task to analyze the resultant net when the modeling of an FMS with more restrictions. And they would pay much more costs to do on-line rescheduling in case of accidents such as machine breakdowns or urgent jobs.

In this paper, we propose a genetic search algorithm based on the timed Petri net model in order to find an optimal or near optimal dispatching rules under specific performance measures and restrictions. Most importantly, a genetic algorithm embedded search based penalty strategy is proposed to find a feasible and near optimal dispatching methods to resolve the conventional problem with exponential growth of search time and the problem size.

This paper is organized as follows. Firstly, we define FMS in Section 2. Section 3 presents a timed Petri net model for an FMS. In Section 4, a GA embedded search method will be employed to solve a dynamic dispatching problem. Finally, conclusions are given in Section 5.

2. FLEXIBLE MANUFACTURING SYSTEMS

This section deals with a flexible manufacturing system in which products are manufactured using several kinds of machines. The entire plant contains load station, unload station, some kinds of working machine, Input/Output buffers. Products, load/unload stations, machines, and buffers have the following features:

1. Products: A product is produced when a job is processed completely in the plant. Each job includes a series of operations in a prescribed order. A half-completed product is obtained when an operation is processed.
2. Materials/products warehouse: Materials warehouse is considered to be an unlimited buffer where unprocessed jobs or some kinds of half-completed products are organized in order. Products warehouse requires unlimited buffers where completed products are stored.
3. Load station: Load station is a transported center where a selected job from warehouse is transported to a buffer of a chosen machine center.
4. Input buffer: Input buffer is a temporary, limited space where a selected job from warehouse or from output buffer of one kind of working machine is stored in order.
5. Working machine: Each job's operation requires a specific type of working machine. A prescribed time called processing time is needed to complete an operation. When a machine processed a job's operation completely, setup time is required to prepare for the next operation.
6. Output buffer: Output buffer is a temporary, limited space where one job's operation is completed from one-kind of working machine. Some half-completed products are stored in output buffer to be choosen and transported to input buffer working machine.

We schematized a FMS in Figure 1. A plant was organized with warehouses, load/unload station, and machine centers. Initially, Jobs enumerated were put to the materials warehouse in natural order. In the figure, Hundreds/thousands of products were manufactured in the plant through a certain stages. These checkpoints are discussed as follows:

stage 1: Choose a not completed job from warehouse by warehouse-job-selection strategy such as first in first out (FIFO), shortest-time remained job first (SRJF), longest-time remained job first (LRJF), shortest-times abandoned job first (SAJF) or longest-times abandoned job first (LAJF). In this stage, a selected job is send to load station.

stage 2: In this stage, one job is then transported to *i-th* machine center from load station using machine-type-selection strategy. A machine-type-selection strategy is such as shortest-cost-needed machine first (SCMF), longest-cost-needed machine first (LCMF), shortest unemployed machine first (SUMF) or longest unemployed machine first (LUMF).

stage 3: In this stage, one job is choosen to be operated from input buffer of machine center by job-from-input-buffer selection strategy. A job-from-input-buffer selection strategy is such as FIFO, SRJF, LRJF, shortest-abandoned job first (SAJF), or longest-abandoned job first (LAJF). An abandoned job is a job which sent back to materials warehouse with uncompleted state.

stage 4: In this stage, two jobs or more are in competitive situation when they are choosed to the input buffer of the same machine center with only one space available. To avoid dead lock in this situation, one strategy such as SRJF, LRJF, SAJF, LAJF, shortest waiting-output-buffer job first (SWOJF), or longest waiting-output-buffer job first (LWOJF) is selected.

stage 5: In this stage, one job is selected to be aborted from a full output buffer of a machine center. Aborted job is choosen under the strategy of SRJF, LRJF, SAJF, LAJF, SWOJF, or LWOJF.

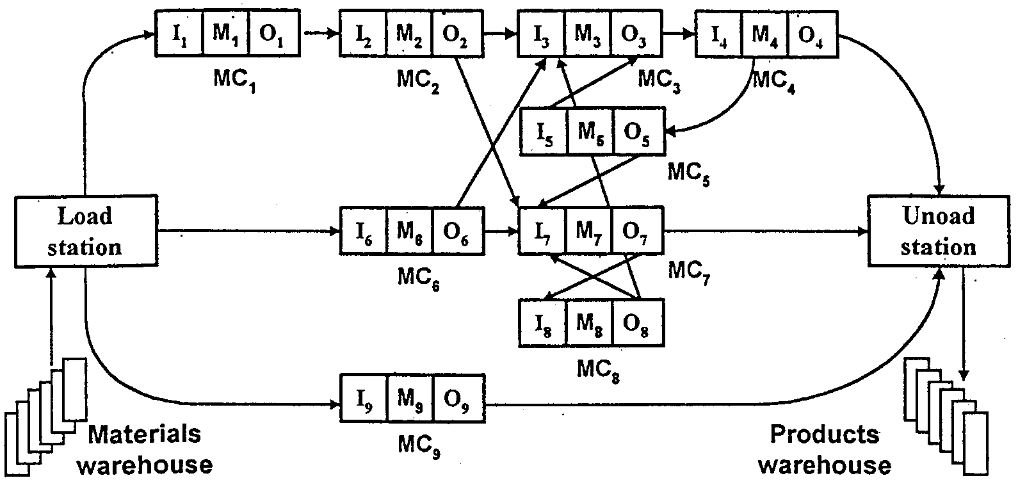

Fig. 1. A plant.

	OP_1	OP_2	OP_3	OP_4	OP_5	OP_6	OP_7	Process Time	Cost spent
Route 1	MC_1	MC_2	MC_3	MC_4	MC_5	MC_3	MC_4	177	70
Route 2	MC_1	MC_2	MC_3	MC_4	MC_5	MC_7	-	57	70
Route 3	MC_1	MC_2	MC_7	MC_8	MC_3	MC_4	-	57	70
Route 4	MC_1	MC_2	MC_7	MC_8	MC_7	-	-	37	70
Route 5	MC_6	MC_7	MC_8	MC_7	-	-	-	33	70
Route 6	MC_6	MC_7	MC_8	MC_3	MC_4	-	-	53	70
Route 7	MC_6	MC_3	MC_4	MC_5	MC_3	MC_4	-	73	70
Route 8	MC_6	MC_3	MC_4	MC_5	MC_7	-	-	53	70
Route 9	MC_9	-	-	-	-	-	-	5	70

Fig. 2. Route of one completed job.

Jobs \ Checkpoint No	1	2	3	4	5	Total time
500	LRJF	LUMF	LRJF	SRJF	SRJF	9261
500	LRJF	LUMF	SRJF	LWOJF	SWOJF	7501
500	LRJF	SUMF	LRJF	LWOJF	LWOJF	8013
500	SAJF	LCMF	LAJF	LAJF	SRJF	5147

Fig. 3. Time to complete 500 jobs using different strategies of checkpoints.

The basic criteria of a plant's throughputs is based on:

1. products enumerated were manufactured completely for a considerable time,
2. elapsed time is counted for a fixed number of products .

Suppose a job is completed using the same fixed cost. A machine center is defined as follows:

Definition 1: Let $P = \{MC_k \mid MC_k = (I_k, O_k, OP_k, St_k, C_k), k = 1, 2, \ldots, n\}$ denote a job is completed by a set of k machine centers with input buffer of size I_k, output buffer of size O_k, operational time unit of OP_k, set-up time unit of St_k, and cost spent of C_k.

In Fig. 1., let $MC_1 = (3, 4, 10, 2, 10)$, $MC_2 = (2, 3, 9, 1, 10)$, $MC_3 = (4, 3, 15, 2, 10)$, $MC_4 = (3, 4, 10, 2, 10)$, $MC_5 = (5, 3, 8, 1, 10)$, $MC_6 = (12, 10, 15, 10, 20)$, $MC_7 = (12, 7, 5, 6, 10)$, $MC_8 = (4, 5, 8, 1, 10)$, $MC_9 = (20, 30, 5, 25, 70)$ denote initial state of each machine center. Thus, one job P_k is completed through a specific route as shown in Fig. 2.

There exist 3,600 combinations when one job was choosed from materials warehouse and operated through some machine centers. In Fig. 3., time elapsed is counted using different strategies of checkpoints to complete 500 jobs.

3. PETRI NET MODELING FOR SCHEDULING

The reader is referred to [3, 5, 6] for tutorials on Petri nets. A Petri net is formally defined as a 6-tuple [3, 5, 9-10]

$N_p = (P, T, Q, I, O, M_0)$,
where

1. $P = \{p_1, p_2, \ldots, p_n\}$ is a finite set of places, and $n \geq 0$;
2. $T = \{t_1, t_2, \ldots, t_m\}$ is a finite set of transitions, and $m \geq 0$;
3. $Q = \{q_1, q_2, \ldots, q_k\}$ is a finite set of state tokens, and $k \geq 0$;
4. $I \subseteq (P \times T) \rightarrow N$ is an input function set that defines a set of directed arcs from places to transitions, where N is a set of non-negative integers, and each p_i is an input place of t_j if $(p_i, t_j) \in I$;
5. $O \subseteq (T \times P) \rightarrow N$ is an output function or weight function set that defines a set of directed arcs from transitions to places, where N is a set of non-negative integers, and each p_i is an output place of t_j if $(t_j, p_i) \in O$; and
6. M_0: $P \rightarrow N$ is an initial marking.

An input and an output sets are defined for a transition. For representing a dynamic system and the behavior of a modeled system, each place can potentially hold either none or a number of tokens. The distribution of tokens over places is called a marking of the Petri net. A token is expressed as a small solid dots. In general, Petri nets have a dynamic life determined by the tokens inside of a place that activate the place in which they appear [7].

The presence or absence of a token indicates whether a condition associated with a place. On the other hand, a transition may fire when every input places of the transition contains at least one token. A Petri net that contains tokens is called a marked Petri net. A marking of a Petri net with n places is represented by an $(n \times 1)$ vector M; elements of the vector, denoted as $M(p)$, are non-negative integers representing the number of tokens in the related places [3]. Since a place represents the availability of resources, the number of tokens in the place represents the number of available resources in the place [5, 8]. The changing distribution of tokens on places represents the occurrence of events in a dynamic modeled system. A scheme of dynamic behavior has been introduced in [5, 6] to deal with the fire mechanism of a transition.

In our timed Petri net model, place symbol $\textcircled{L}$, $\textcircled{I}$, $\textcircled{O}$, $\textcircled{St}$, and $\textcircled{U}$ denote load station, input buffer, output buffer, setup time, and unload station, separately. Load station is enabled if marterials warehouse is not empty. Transition connected to $\textcircled{L}$ is fired if input buffer of machine center is not full. Machine in machine center operates while it's input buffer is not empty and setup time is ready. Fig. 4. is our timed Petri net model for FMS scheduling as shown in Fig. 1.

4. GA BASED DISPATCHING ON TIMED PETRI NET

Genetic algorithms (GAs) are a set of stochastic algorithms that represent a rich and not very complicated subject matter for the investigation of complex optimization problems. GA was developed by Holland [11] to mimic the behavior of natural evolution and to design the artificial system software that imitates the adaptive mechanism of the natural systems. It has been shown that GA can be applied to general NP-complete problems [12].

GAs are population-based search algorithms. They start with a set of contending trial solutions called initial population. The trial solutions maintained in population are termed chromosomes and are encoded into bit string in which each bit is termed a gene. Each gene stands for a chooseable method as SCMF method in machine-type-selection strategy.

Chromosomes are competitive to survive within the evolutionary generation to optimal solutions. Genetic operators, i.e., selection, crossover, and mutation, are employed to alter the population. The fitness of each trial solution is assessed by the measure of fitness function.

GAs can be represented as a set of parameters

$G = (E, op, P, F, T, N)$,

where

1. E is an encoding schema to map domain knowledge to codes;
2. op is the genetic operators applied to evolutionary search;
3. P are the probabilities of control frequencies of genetic operators;
4. F is the fitness function to evaluate the fitness value of chromosomes;
5. T is the terminated condition; and
6. N is the population size.

The performance of GAs varies with the implementation parameters. Fig. 5 illustrates flowchart of a standard GA. Given an initial population, i.e., a set of random chromosomes, GA uses operators to select or alter chromosomes in population until a predetermined stop condition is satisfied.

Conflicted places in out timed Petri net are defined as more than one token inside them or more than output transitions connected to them. The conflicted places are referred to checkpoints. Thus, a combination of checkpoints is a set of chromosome. A chromosome is represented schematically in Figure 6.

The GA based dispatching rules on timed Petri net is now presented as follows:

GA based dispatching (GABD) rules on timed Petri net algorithm

```
Input: number of jobs, state of machine center
       (input buffer, setup time, output buffer,
       operation time, cost), penalties of job.
Output: complete time of total jobs
begin
  Set each value of generation, population,
      chromosomes
  for $j \leftarrow 1$ to numbers of generation
    begin
    for $k \leftarrow 1$ to numbers of chromosome
      begin
        Generates initial chromosomes
          randomly, each chromosome
          consists of combinations of
          checkpoints.
        while materials warehouse is not
            empty
          Record completed time of each
              unevaluated chromosome
        end while
      end $j$
      Reproduct the best one
      Generate the best half of chromosomes
          minus one by crossover
      Randomized generating the rest of
          chromosomes
    end $k$
  Output the best chromosome
end.
```

Based on the proposed algorithm, five hundred jobs completed using 4,437 time units as shown in Fig. 7. In this state, job is choosen by LAJF method from materials warehouse, by LUMF method to select a machine center, by SAJF method from input buffer. One job is to give back a job to materials warehouse by LWOJF in case of two jobs or more compete for machine center with only one input buffer available, and by SRJF from a full buffer.

In Fig. 8, Y-axis is the time unit exhausted when completing the 500 jobs. X-axis represents different chromosomes (sorted). In the third generation, almost the near-optimal solution is found in our example of Fig. 1.

5. CONCLUSION

In this paper, a FMS dispatching strategies are considered. Firstly, we use a timed Petri net for modeling FMS scheduling. Conflicted places in our timed Petri net are defined as confliction of resources assignment and resources dispatching order. The conflicted places are referred to checkpoints. A combination of checkpoints is a set of chromosome. Owing to the NP-complete nature, the chromosomes use genetic operators to change information in next generation to get the optimal or sub-optimal solution. The fitness associated with penalties of each trial solution is assessed by the measure of fitness function. Our experiments show that the proposed approach does represent a good alternative for dispatching strategies of FMS problems.

REFERENCES

[1] Y-F Chiu, and L-C Fu, "A GA embedded dynamic search algorithm over a Petri net for an FMS scheduling," *in Proc. of 1997 IEEE Int. Conf. on Robotics and Automation,* pp. 513-518, 1997.

[2] Mu Der Jeng, "A Petri net synthesis theory for modeling flexible manufacturing systems," *IEEE Trans. Systems, Man, and Cybernetics,* Vol. 27, No. 2, pp. 169-183, 1997.

[3] Chen, S. M., J. S. Ke, and J. F. chang, "Knowledge representation using fuzzy petri nets," *IEEE Trans. Knowledge and Data Engineering,* Vol. 2, No. 3, pp. 311-319, 1990.

[4] Sun, T. H., C. W. Cheng, and L. C. Fu, "A Petri net based approach to modeling and scheduling for an FMS and a case study," *IEEE Trans. Industrial Electronics,* Vol. 41, No. 6, pp. 593-601, 1994.

[5] Zurawski, R. and M. Zhou, "Petri nets and industrial applications: A tutorial," *IEEE Trans. Industrial Electronics,* Vol. 41, No. 6, pp. 567-583, 1994.

[6] T. Murata., "Petri nets: Properties, analysis, and applications," *Proc. IEEE,* Vol. 77, No. 4, pp. 541-580, 1989.

[7] Looney, C., "Expert control design with fuzzy rule matrices," *Int. J. Expert Syst.:Res. Appl.,* Vol. 1, No. 2, pp. 159-168, 1988.

[8] Looney, C., "Fuzzy Petri nets for rule-based decisionmaking," *IEEE Trans. Systems, Man, and Cybernetics,* Vol. 18, No. 1, pp. 178-183, 1988.

[9] Pedrycz, W. and F. Gomide, "A generalized fuzzy Petri net model," *in Proc. IEEE,* Vol. 77, No. 4, pp. 541-580, 1989.

[10] Peterson, J. L., *Petri Nets, Theory and the Modelling of Systems,* Prentice Hall, Englewood Cliffs, NJ: Prentice Hall, 1981.

[11] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Reading, MA: Addison-Wesley, 1989.

[12] L. Davis, *Genetic algorithms and simulated annealing,* Morgan Kaufmann, 1990.

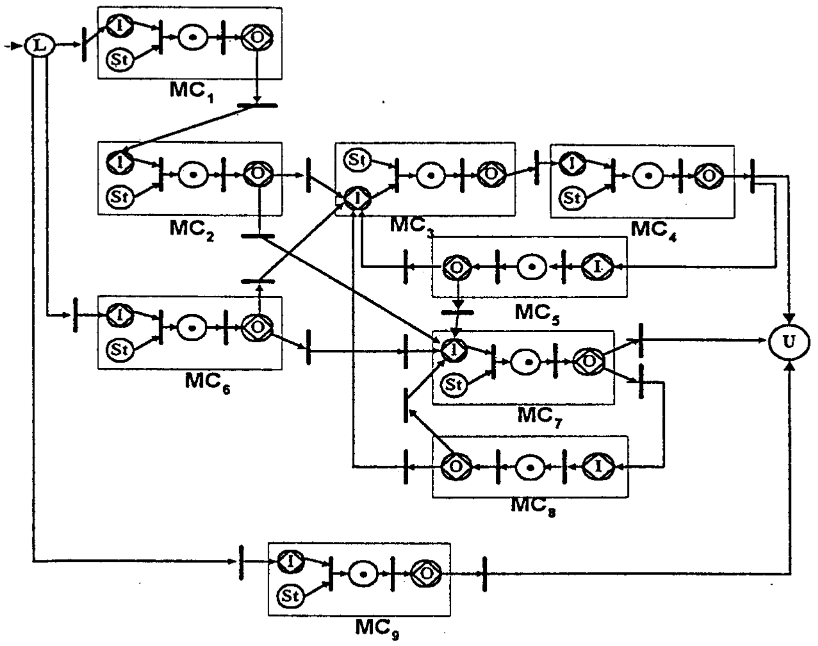

Fig. 4. Our timed Petri net model for FMS scheduling as shown in Fig. 1.

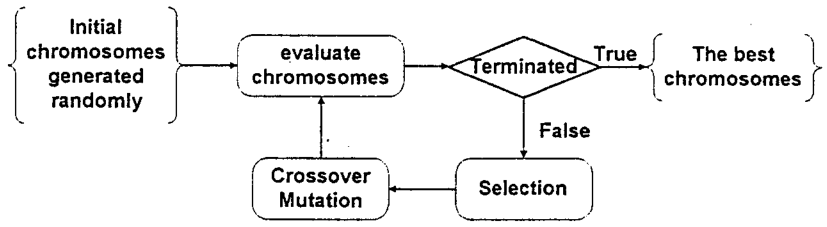

Fig. 5. Flowcharts of GA

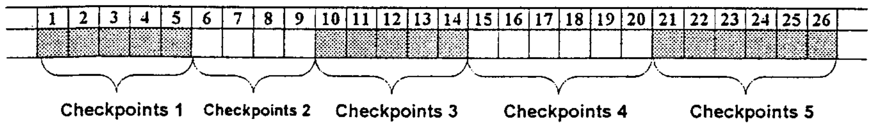

Fig. 6. A chromosome

Checkpoint No / Jobs	1	2	3	4	5	Total time
500	LAJF	LUMF	SAJF	LWOJF	SRJF	4437

Fig. 7. Five hundred jobs were completed using 4,437 time units based on GABD.

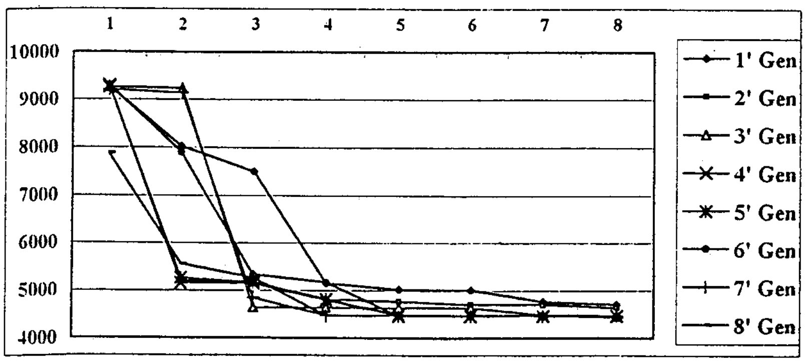

Fig. 8. Time unit spent with different generations.