Automation and Robotics in Construction XVI © 1999 by UC3M

PARALLEL FLOW SHOPS SCHEDULING PROBLEM USING GENETIC ALGORITHMS

Hali Pang Binglin Zheng Xinhe Xu

School of Information Science & Engineering, Northeastern University
Shenyang (110006) P. R. China

Abstract: In this paper, a general parallel flow shops problem with the objective of minimizing total earliness and tardiness has been addressed. In view of the intractable nature of the problem, a Genetic Algorithms (GAs) based approach is proposed to solve the problem. This algorithm has been tested on some randomly generated test problems. Computational results show the proposed approach is quite heartening.

Keywords: Parallel Flow shops, Scheduling ,Total earliness and tardiness, Genetic algorithms

1. INTRODUCTION

Parallel machine and flow shop scheduling problems have been studied intensively, and some valuable results have been reported in the research literature [1-4]. However, as an extension of above the problem, the problem of scheduling jobs on parallel processing systems where the systems are flow shops called Parallel flow shops has received little attention. This problem is a special case of general multi-stage multi-machine problem known as Hybrid flow shop. Sundararaghavan et al. [5] addressed a simple version of the parallel flow shops scheduling problem, and proposed heuristic algorithm for the objective of minimizing makespan. In their paper, the problem researched is limited to two flow shops where each flow shop include only two machines, and a situation that processing time of jobs on corresponding machines belong to different flow shops are proportional to each other is considered.

In this paper, we consider a more complex criterion and situation: there are several flow shops where each flow shop consist of the same number of machines in series, and processing time of different flow shops are unrelated. The objective is to minimize total earliness and tardiness. This problem can be find in some industrial environment. A practical example in construction is given in next section. The rest of this paper is organized as follows: In section 3, the problem description and formulation are given. Section 4 present an implementation of the GAs, and Section 5 discusses the experimental results following conclusions of this research in Section 6.

2. PARALLEL FLOW SHOP: A RACTICAL EXAMPLE IN CONSTRUCTION

A typical construction engineering involves some sub-projects such as main body, subsidiary devices installation and decoration etc. In each sub-project, there are many processes or stages which interlink each other in time and space to form a very complex network. There are some key processes or stages in the network, from view of the whole, the whole construction process is composed of these key stages successively.

Consider a construction firm which include a management department and several units in charge of construction called teams. The team can complete any project independently, because it consists of variety kinds of employee and construction equipment which have been classified based on different type of work clearly. When a certain stage of a project is finished, the employee and machines in this stage can turn to corresponding work in other project. The duty of the management department are to sign construction engineering contracts, assign the contracts to each team and supervise the engineering progress.

Assuming that there are n projects at a certain start time. The problem is to assign the projects to the teams and sequence them to utilize resources efficiently, satisfy the due date and pursue maximum benefit (or minimum cost). Minimization of total earliness /tardiness objective function is often used to ensure the objective of the firm. At each stage j, different time is needed to accomplish the work for different team, because the employee and equipment included in different teams are different. So, if we looked the contract as a job, looked the team and key stage as a flow shop and machine respectively, the situation can be viewed as a unrelated processing time parallel flow shop. Fig.1 presents a schematic diagram of this scheduling situation with three stages.

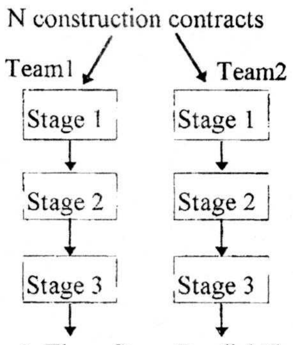

Figure 1. Three Stage Parallel Flow Shop

3. THE PROBLEM DESCRIPTION AND FORMULATION

Parallel Flow shops scheduling problem can be stated as follows: there are K flow shops where each flow shop consists of M machines in series, N different jobs where each job must be processed by the M machines of any flow shop in the same order. All jobs are available at time t=0, and there is no job passing. Each job is assigned to just one flow shop, which means the job has to be processed in the same flow shop, once the assignment is made. The objective is to minimize total earliness and tardiness cost.

The mathematical model, adequately describing the problem, is given bellow. The notation of all the variables and parameters used are defined firstly, and then the model is presented in the form of an objective function and its associated constraints.

Notation

i, j= index of jobs, i, j= 1, 2, ..., N.
k= index of flow shops, k=1, 2, ..., K.
m= index of machines in a flow shop, m= 1, 2, ..., M.
P_{ikm} = processing time for m-th operation (on m-th machine) of job i in flow shop k.
C_{ikm} = completion time for m-th operation of job i in flow shop k.
S = a big positive number.
d_i = due date of job i.
α_i = earliness penalty of job i per unit time early.
β_i = tardiness penalty of job i per unit time tardy.
E_i = earliness of job i.
T_i = tardiness of job i.
X_{ik} = 1 if job i is processed in flow shop k, and 0 otherwise
Y_{ijkm} = 1 if job i precedes job j on machine m in flow shop k, and 0 otherwise.

The problem can be formulated ad follows:

$$\text{minimize} \quad \sum_{i=1}^{N}(\alpha_i E_i + \beta_i T_i) \qquad (1)$$

subject to:

$$\sum_{k=1}^{K}\sum_{i=1}^{N} X_{ik} = N \qquad (2)$$

$$\sum_{k=1}^{K} X_{ik} = 1 \qquad i = 1,2 \cdots N \qquad (3)$$

$$C_{jkm} - C_{ikm} + (3 - X_{ik} - X_{jk} - Y_{ijkm})\cdot S \geq P_{jkm}$$
$$i,j = 1,2 \cdots N; k=1,2 \ldots K; m = 1,2 \cdots M \qquad (4)$$

$$C_{ikm} - C_{jkm} + (2 - X_{ik} - X_{jk} + Y_{ijkm})\cdot S \geq P_{ikm}$$
$$i,j = 1,2 \cdots N; k=1,2 \ldots K; m = 1,2 \cdots M \qquad (5)$$

$$\sum_{k=1}^{K} X_{ik} \; (C_{ikm+1} - C_{ikm} - P_{ikm+1}) \geq 0$$
$$i = 1,2 \cdots N \; ; \; m = 1,2 \cdots M\text{-}1 \qquad (6)$$

$$E_i = \max\{d_i - \sum_{i=1}^{K} X_{ik} C_{ikM}, 0\} \qquad (7)$$

$$T_i = \max\{\sum_{i=1}^{K} X_{ik} C_{ikM} - d_i, 0\} \qquad (8)$$

$$X_{ik} = 0,1 \qquad (9)$$

$$Y_{ijkm} = 0,1 \qquad (10)$$

The model is formulated as a mixed integer programming model. The objective function focuses on minimizing total earliness and tardiness cost. Constraint (2) demonstrate all jobs must be processed in the flow shops. Constraint (3) describe that each job can only be assigned to just one flow shop. The requirement that no two operations of two different jobs can be processed on the same machine at a time are guaranteed by Constraints (4) and (5). They state that if the m-th operation for job i in flow shop k preceded the m-th operation for job j, then the m-th operation for job j can only start after the m-th operation for job i, and vice versa. Constraint (6) describe that each job must be processed on M machines successively. Constraints(7) and (8) allow that the objective function value can be calculate.

This problem is quite intractable, because the objective function is non-regular. Even the simplest model that involve the earliness and tardiness has been proved to NP_complete[6]. Genetic algorithms, which are developed based on the mechanisms of evolution, demonstrated their potential for solving hard intractable optimization problems[7,8]. Therefore, an approach based genetic algorithms is developed to solve the problem.

4. GENETIC ALGORITHMS

Genetic algorithms are probabilistic search techniques developed based on the mechanism of evolution. The search procedure in GAs, combined with reproduction and recombination. In GAs the solution space is generally represented by a population of structures, where each structure called chromosome, in general, is a possible solution to the problem. From the concept of genetics that better parents produce better offspring, new chromosomes (offspring) are generated by applying genetic operators such as crossover, mutation, and inversion to the potential (parent) chromosomes selected from the existing population. The members with higher fitness values in the current population will have higher probability of being selected as parents. In every generation, the existing population of parent solutions is replaced with newly generated population of offspring solutions. This procedure is repeated to perform an adaptive search, the algorithms converge to the best chromosome, which hopefully represents the optimal or near optimal solutions. Let P(t) be the parents and C(t) be the offspring in the current generation t, the overall procedure of genetic algorithm is show as follows:

```
procedure: genetic algorithms
    begin
      t←0
      initialize P(t)
      evaluate P(t)
      while (not termination condition) do
         recombine P(t) to yield C(t)
         evaluate C(t)
         select P(t+1) from C(t)
         t←t+1
      end do
    end
```

4.1 Representation

There are two essential issues to be dealt for parallel flow shops scheduling problems:

- partition jobs to flow shops
- sequence jobs for each flow shop

The representation designed to encode these two things into a chromosome consists of a job symbol list and a partitioning symbol list, in which integers are used to represent all possible permutation of jobs (or sequence of jobs) and asterisks ※ are used to designate the partition of jobs to flow shops. Let us consider a simple example of the problem with 9 jobs and 3 flow shops. The chromosome can be represented as follows: [2 5 6 ※ 3 1 4 ※ 9 7 8]

This chromosome stands for job 2, 5 and 6 are assigned to flow shop1, and processing sequence is 2→5→6. Generally, for an n-job k-flow shop problem, a legal chromosome contains n job symbols and k-1 partitioning symbols, resulting in total size of (n+k-1).

4.2 Genetic operators

Crossover and mutation are two genetic operators commonly used in the genetic algorithms. Usually, the crossover is used as main genetic operator and the performance of a genetic system depends, to a great extent, on the performance of the crossover operator used; while the mutation is used as a background operator, which produces spontaneous random changes in various chromosomes.

As mentioned, the essential issue of parallel flow shops scheduling problem is the partition and permutation of the jobs in flow shops. Both crossover and mutation operators are designed to handle the job partition and job permutation.

Crossover

We have designed a subschedule preservation crossover operator because we consider subschecule to be the natural building blocks. The subschedule here means a complete schedule for one flow shop. The proposed crossover perform with main three steps:

(1) Obtain asterisk positions (overall partitioning structure) from one parent.
(2) Obtain a randomly selected subschedule from the same parent.
(3) Obtain remaining jobs from the other parent by making a left-to right scan.

The operation of crossover is illustrated in Figure 2. We can know that the proposed crossover can adjust job partition and job order simultaneously from Figure 2.

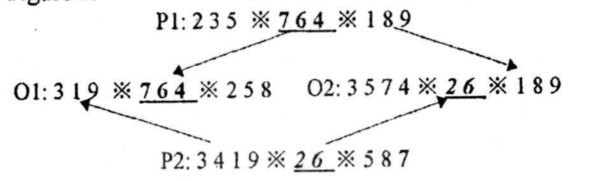
P1: 2 3 5 ※ 7 6 4 ※ 1 8 9
O1: 3 1 9 ※ 7 6 4 ※ 2 5 8
O2: 3 5 7 4 ※ 2 6 ※ 1 8 9
P2: 3 4 1 9 ※ 2 6 ※ 5 8 7

Figure 2. Illustration of Crossover Operator

Mutation

The swapping mutation, which select two random positions and then swap their genes, is used here. The randomly swapped genes may be either job or asterisk. The possible combinations of job and asterisk result in four basic types of mutation.

(1) If both genes are job, and these two selected jobs are processed by the same flow shop. In such case the mutation alters the job order for the flow shop.
(2) If both genes are job, and these two selected jobs are processed by different flow shops. In such case the mutation alters both job order and job partition to flow shops for chromosome.
(3) If both genes are asterisk, the mutation alter job partition only.
(4) If one gene is asterisk and another is job, the mutation alters both jobs order and jobs partition to flow shops for the chromosome.

Mutation is the only genetic operation which can alter the position of asterisks. Without mutation, genetic starch will be confined by initial population (or initial position of asterisks). So mutation plays a vital role in our genetic algorithm implementation.

4.3 Evaluation and selection

The objective function value for each chromosome must convert into the fitness value because the objective is minimization. Let F_i and f_i is the fitness and objective function value of chromosome i respectively, $\bar{f}$ is the average function value of all chromosomes, the procedure can be simply done by the following equation: $F_i = (\bar{f} / f_i)^2 \cdot \bar{f}$.

We use roulette wheel as the selection mechanism to reproduce the next generation, combined with the elitist way. The fitter chromosome has a large chance to be reproduced in to next generation, and the best chromosome can be put into next generation directly.

5. COMPUTATIONAL RESULTS

The proposed GAs has been test on problems with $n \in \{10,20\}$, $k \in \{2,3\}$ and $m \in \{5,7\}$ generated randomly. For each test problem, an integer processing time P_{ikm} from the uniform distribution [1 , 20] was generated for each job i and each machine for each flow shop. The due date of each job was generated from the uniform distribution $[1, \frac{0.6}{K} \max_k \{\sum_{i=1}^{N} \sum_{m=1}^{M} P_{ikm}\}]$. To simplify, we assume $\alpha_i = \beta_i = 1$.

In general, the performance of the GAs depend on the parameters setting to a large extent. Therefor, the suitable parameters for the problems will be determined. First, we investigated suitable population size for each dimension of the problem. Fix the max_gen as 300, and P_m and P_c as 0.1. Under the condition of lower ratio of crossover and mutation, the population size becomes one of the leading factor for GAs. Figure 3 show the relative results under different population size for some test problems, where relative results are ratio of result obtained to the best result. From the results we can see that the factor of population size is relative to the dimension of problem, especially the number of flow shops closely.

We further studied performance of the algorithm under the different parameters of P_m and P_c . Take the population size as suitable value determined above. Run the algorithm only with crossover (P_m =0, and P_c from 0.1 to 0.9) or mutation (P_c =0, P_m from 0.1 to 0.9). In this case, the genetic operator used become the only factor to affect the performance of the algorithm. In Figure 4, the relative objective value of some test problem are potted as a function of the parameter P_m or P_c .

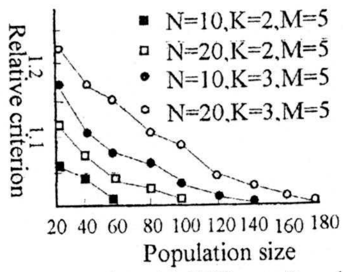

Figure 3. Results with Different Population Size

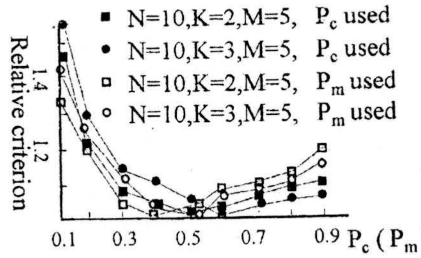

Figure 4. Results with Different P_c or P_m

For all test problem, the parameter of algorithm are set suitable value according to the experiments mentioned above, terminal criterion of the algorithm is maximum generation = 300. We run the algorithm for 10 times on each problem, and corresponding computational results are given in table 1.

Table 1: Computational results

N × K × M	Best solution	Average deviation	Cup time (s)
10 × 2 × 5	98	10.5	3.18
10 × 3 × 5	75	8.2	4.33
10 × 2 × 7	116	11.3	3.56
10 × 3 × 7	104	5.8	4.86
20 × 2 × 5	163	12.7	3.77
20 × 3 × 5	141	9.6	5.12
20 × 2 × 7	168	13.4	4.54
20 × 3 × 7	172	15.2	5.78

6. CONCLUSION

In this paper, a general parallel flow shops problem with the objective of minimizing total earliness and tardiness has been formulated. A genetic algorithms based approach has been developed. Extensive computational experiments on some randomly generated test problems have been performed, and the results are encouraging.

REFERENCES

[1] A. Doframaci and J. Surkis. "Evaluation of a heuristic for scheduling independent jobs on parallel identical processors." Mgmt. Sci., Vol. 25 pp. 1208-1216. 1979.

[2] Li, C. And T. Cheng. "The parallel machine minmax weigthed absolute lateness scheduling problem." Naval Research Logistics, Vol. 41 pp. 33-46. 1993.

[3] M. Garey and D. Johnson. "The complexity of flow shop and jobshop scheduling. " Mach O.R., Vol. 1 pp. 117-129. 1976.

[4] M. Widmer and A. Hertz. "A new heuristic method for the flow shop sequencing problem." Eur. J. Opl. Res., Vol. 41 pp. 186-193. 1989.

[5] P. Sundararaghavan, A. Kunnathur and I. Viswanathan. "Minimizing makespan in parallel flow shop." J. Opns. Res. Soc., Vol. 48 pp. 834-842. 1997.

[6] T.C.E. Cheng and M. C. Gupta. "Survey of scheduling research involving due date determination decisions." Eur. J. Opl. Res., Vol. 38 pp. 156-166. 1989.

[7] D. Goldberg. Genetic Algorithms in Search, Optimization, and Machine learning. Addison-Wesley, Reading, MA. 1989.

[8] D. Lawrance. Handbook of Genetic Algorithms. Van Norstrand Reinhold, New York. 1991.