한국지능정보시스템학회
02 춘계학술대회 논문집, pp. 429~434

다단계 제품 구조를 고려한 유연 잡샵 일정계획의 Large Step Optimization 적용 연구

장양자* · 김기동** · 박진우*

Large Step Optimization Approach to Flexible Job Shop Scheduling with Multi-level Product Structures

Yang-Ja Jang* · Kidong Kim** · Jinwoo Park*

Abstract

For companies assembling end products from sub assemblies or components, MRP (Material Requirement Planning) logic is frequently used to synchronize and pace the production activities for the required parts. However, in MRP, the planning of operational-level activities is left to short term scheduling. So, we need a good scheduling algorithm to generate feasible schedules taking into account shop floor characteristics and multi-level job structures used in MRP. In this paper, we present a GA (Genetic Algorithm) solution for this complex scheduling problem based on a new gene to reflect the machine assignment, operation sequences and the levels of the operations relative to final operation. The relative operation level is the control parameter that paces the completion timing of the components belonging to the same branch in the multi-level job hierarchy. In order to revise the fixed relative level which solutions are confined to, we apply large step transition in the first step and GA in the second step. We compare the genetic algorithm and 2-phase optimization with several dispatching rules in terms of tardiness for about forty modified standard job-shop problem instances.

Key words : Flexible Job Shop, Genetic Algorithm, Large Step Optimization

1. Introduction

The so-called MRP (Material Requirement Planning) is frequently used to synchronize the production activities of a multi-stage manufacturing system. It determines the appropriate quantities and due dates of each component to be produced or procured and releases production or purchase orders to shops. Since MRP does not consider the capacity of the shop level resources and utilizes a fixed lead-time, it frequently generates infeasible plans. So in a well run MRP shop, the MRP plan is usually reinforced by other means such as detailed capacity planning and scheduling tools. In spite of such effort, since the job shop environment is so complex and the number of jobs so large, the machine loading and operation sequencing decisions are frequently done using dispatching rules based on the due dates and processing times of the operations.

Finding an optimal solution in a reasonable time for practical sized scheduling problems with multi-level job structures still remains to be resolved and search solutions such as genetic algorithm, simulated annealing, tabu search and fuzzy logic are increasingly used to cope with large computational requirements posed by realistic scheduling problems(Roach and Nagi, 1996; Anwar and Nagi, 1997; Kimms, 1999; Park and Kim, 2000). Our current study on flexible job shop problem with multi-level job structures is one of such efforts to obtain a workable solution to a practical problem in a reasonable computational time without guaranteeing an optimal solution.

The remainder of the paper is organized as follows. A description of the flexible job-shop scheduling problem with multi-level job structures is given in section 2, the proposed GA approach in section 3, and the numerical experiments in Section 4. Finally, conclusions and additional remarks are presented in section 5.

2. Problem Description

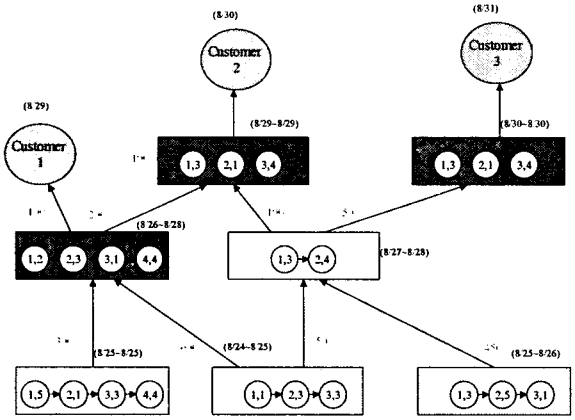
Most of the past and current research literatures related to job-shop scheduling problems assumes simple string type and mutually independent jobs. In reality, there are not only precedence relationships between jobs but also resource flexibility and non-linear routings. The resource flexibility can be defined as the existence of alternative routings or machines. The non-linear routing reflects a cycle in the manufacturing process including rework and reentry, as well as tree-like processes such as assembly or disassembly. The job shops featuring resource flexibility are called either flexible or multi-resource job shops. Also, a job with preceding jobs can start only after the completion of the preceding jobs. Although real shops possess all the characteristics mentioned above, most previous scheduling researches have simplified these complex characteristics to reduce problem complexity.

During the MRP planning process, the quantities and time

* 서울대학교 산업공학과
** 강원대학교 산업공학과

slots for the production of end items and components are determined taking into account of the multi-level product structures and routing. Therefore the mixed form of assembly and disassembly trees and the production orders indicating quantities and production time-slots are the inputs to the scheduling problem we are dealing with (Figure 1).

Figure 1. Example of production orders



2.1. Problem formulation

This research aims to obtain production schedules that will maintain the MRP production order due dates wherever possible. We also consider precedence relationships between operations as well as between jobs. An operation of a job may be ready when all materials are procured and all the preceding operations have been finished. The formulation takes into account multiple identical parallel machines that are grouped into a work center. Furthermore, each machine has a machine calendar showing the time intervals of machine unavailability. We create and insert dummy operations to deal with unavailable time intervals.

Let us first introduce some of the notations used. The known parameters are as follows:

N : Number of jobs
i : Index of a job
ij : operation j of job i
pq : operation q of job p
n_i : Number of operations of job i
D_i : Due date of job i
R_i : Ready time of job i
S_i : Set of subsequent jobs of job i
F : Set of first processed jobs
W : Number of workcenters
w : Index of a workcenter
m, m' : Index of a machine
M_w : Set of parallel machines in workcenter w
t_{ijw} : Processing time of operation j of job i
O_w : Set of operations processed in workcenter w

The decision variables are:

$$X_{ijwm} = \begin{cases} 1 \text{ If operation } ij \text{ may be processed on machine } m \\ \quad \text{, where } m \in W_m \\ 0 \text{ Otherwise} \end{cases}$$

$$Y_{ijpq} = \begin{cases} 1 \text{ If operation } ij \text{ precedes operation } pq, \\ 0 \text{ Otherwise} \end{cases}$$

S_{ijwm} : start time of operation j of job i
C_{ijwm} : completion time of operation j of job i
T_i : tardiness of job i

Using these notations, we are now able to present the following MIP (Mixed Integer Programming) model formulation.

Objective function (1) minimizes the total delay. Equation (2) defines the job tardiness and (3) constrains the duration of an operation to a value equal to the processing time. Constraints (4) and (5) guarantee the precedence relationships among operations. Equation (6) constrains the ready time of each job. Equation (7) ensures that operation ij is processed on one of the identical parallel machines designated as M_w in work center w. Equation (8) describes a disjunctive constraint between ij in a set which results from O_w excluding last operations rs and pq, and in a set which results from O_w excluding ij. If Y_{ijpq} is 1, then the condition $C_{pq} - C_{ij} \geq t_{pqw}$ has to be satisfied, because ij

$$\min \sum_{i=1}^{N} T_i \qquad (1)$$

s.t.

$$T_i - C_{in_i} \geq -D_i \qquad \forall i \qquad (2)$$

$$C_{ij} - S_{ij} = t_{ijw} \qquad \forall m \in M_w; \forall ij \in O_w; \forall w \qquad (3)$$

$$S_{ij} - C_{i(j-1)} \geq 0 \qquad i = 1,...,N; j = 2,...,n_i \qquad (4)$$

$$S_{pl} - C_{in_i} \geq 0 \qquad p \in S_i; \forall i \qquad (5)$$

$$S_{il} \geq R_i \qquad \forall i \qquad (6)$$

$$\sum_{m \in M_w} X_{ijwm} = 1 \qquad \forall ij \in O_w; \forall w \qquad (7)$$

$$\left.\begin{aligned} & X_{ijwm} + \sum_{m' \neq m} X_{pqwm'} + Y_{ijpq} \leq 2 \\ & C_{pq} - C_{ij} + M(3 - Y_{ijpq} - X_{ijwm} - X_{pqwm}) \geq t_{pqw} \\ & C_{ij} - C_{pq} + M(2 + Y_{ijpq} - X_{ijwm} - X_{pqwm}) \geq t_{ijw} \end{aligned}\right\} \quad \begin{aligned} & \forall ij \in O_w - \{rs\}; \forall pq \in O_w - \{..., ij\} \\ & , where\ O_w = \{..., ij, ..., pq, ..., rs\} \\ & \forall m, m' \in M_w \end{aligned} \qquad (8)$$

is completed before pq.

Our model is more complex than the standard job-shop scheduling problem because of added features such as machine selection and precedence constraints. A more efficient search algorithm is needed to yield a good feasible solution.

3. Solution methodology

Our problem consists of three sub-problems: the assignment problem of selecting one machine among parallel machines in each work center, the sequencing problem of determining the order of the operations and the timing problem which determines the operation start time.

The genetic algorithm is known as one of the meta-heuristic method that provides a relatively good solution for problems of high complexity. We propose a GA approach in this section. The implementation of the GA requires following fundamental phases:

- Modeling of the real problem by means of chromosome representation
- Defining a suitable fitness function
- Defining suitable genetic operators
- Defining how to retain solution feasibility

3.1. Solution encoding based on problem characteristics

Most existing approaches to flexible job shop scheduling have been hierarchical, that is, they first determine machine assignment and then sequence the operations. The hierarchical approach may be based on the observation that when a machine is chosen to process an operation, flexible job shop scheduling becomes the classical job shop problem. Given the machine assignment, the problem is to find the sequence of operations which minimize a given performance function. A representative research based on the hierarchical approach by Brandimarte (1993), involved the sequencing of operations firstly by dispatching rules and then the assignment of operations to machines. Alternatively, the machine assignment problem may be solved by a load-balancing method, and then operations sequenced afterwards. The simultaneous assignment and sequencing approach can be found in the work of Mastrolilli and Gambardella (2000).

In our approach, we tackle the problem simultaneously by inventing a gene designed to reflect the precedence relationships of operations. We invent the Relative Operation Level (ROL) that incorporates the level information and randomness to control the pace of the branches dynamically.

Gene (o_{ij}, m_{ij}, l_{ij}) : Operation o_{ij} may be processed in machine m_{ij} and has a level value l_{ij}.

Level l_{ij} determines operation sequence and has the following property:

Property 1: Let (o_{ij}, m_{ij}, l_{ij}), (o_{pq}, m_{pq}, l_{pq}) represent genes of o_{ij} and o_{pq}, respectively. Then if $l_{ij} < l_{pq}$, o_{ij} precedes o_{pq} ($o_{ij} \prec o_{pq}$).

3.2. ROL setting procedure

After generating the random number r_{ij} for all operations, the interval of an operation is calculated by equation (9) shown below. The scale factor in (9) simply converts ROL to an integral value.

$$Interval\ of\ o_{ij} = \frac{r_{ij}}{\sum r_{ij}} \times (Number\ of\ Remaining\ Operations\ of\ o_{ij}) \times (Scale\ Factor) \quad (9)$$

Given the level value of the final operation at random, we set ROL values of all operations backward using the interval of the operation and equation (10).

$$ROL_{ij} = ROL_{i,j+1} - Interval\ of\ o_{ij} \quad (10)$$

With this chromosome, we can set all the 0/1 variables defined in section 2, and the MIP model can be transformed into an LP (Linear Programming) model. When generating an initial solution, we select the machine to process each operation, and then sequence the operations in the order of increasing operation due date.

3.3. Genetic Operators

An operation sequence is defined as a gene as in other job shop scheduling studies, and we use simple two-point crossover and bit mutation method among various genetic operators that have been proposed. The two-point crossover selects two levels at random, and then generates two children by crossing over the information from two parents. Each child merges the partial operation sequences from the two parents and forms a new sequence. The bit mutation is used as a mutation operator, which selects an operation at random and then changes the assigned machine.

3.4. Evaluation of chromosome

We use the roulette wheel approach as a selection procedure. To select a good chromosome, we have to evaluate the fitness of each chromosome. We simply use the total delay as the selection criterion. As mentioned earlier, the MIP model in section 3 can be transformed into the LP model by applying chromosome information. Hence, we can use the LP solver to evaluate each chromosome. However , the evaluation time increases as the problem size increases in the use of the LP solver. So, we use the forward scheduling algorithm from the ready time of each job to the higher level operations. The overall genetic algorithm procedure is as follows:

Procedure Genetic Algorithm:

While
    Begin
    For 1 to ROL Iteration

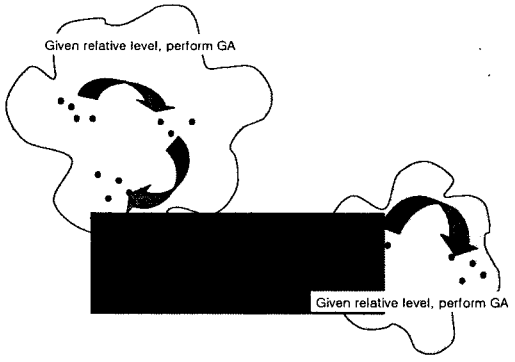
```
Begin
 Set ROL();
 Generate Initial Solution();
 Evaluate();
 For 1 to Generation No
     Begin
     Select();
     CrossOver();
      Mutate();
      Evaluate();
      End
 End
End
```

3.5. Large step optimization

Lourenço(1995) introduced a new randomized methods, designated as large step optimization methods, in the job shop scheduling, where at each iteration a large step is performed, followed by a local optimization method. These methods only consider local optimal solutions and therefore the solution space is reduced Also, they have the property of being able to make large changes in the current solution and thus getting out of possible valleys of the cost function.

In order to revise the fixed relative level which solutions are confined to, we apply large step transition in the first step and GA in the second step shown in Figure 2.

Figure 2. Large step optimization



The three main routines of a large step optimization method are: the method to perform a large step, the one for the small steps and the accept/reject test. The accept/reject test can accept only downhill moves.

The overall large-step optimization procedure is as follows:

Procedure Large-step Optimization:

Step1. Get an initial schedule σ_0

Step2. Perform the following loop numiter times:

Large step transition: change levels, $l_i \rightarrow l_{i+1}$

Small steps: run a GA with l_{i+1} and obtain σ_{i+1}

Perform an accept/reject test comparing σ_i with σ_{i+1}

Step 3. Return the best solution found σ_{best}

4. Computational experiments

For the computational experiments, we reconstruct the standard job shop problems LA01 to LA40 presented by Lawrence (1984) into problems with multi-level job structures. The numbers of jobs are 10, 15, 20 and 30, and the numbers of total machines are 5, 10 and 15, in Lawrence's original problem. We set the job levels to 2,3 and 4 and the work centers to 3,4 and 5, so that the total machines are categorized into work centers at random.

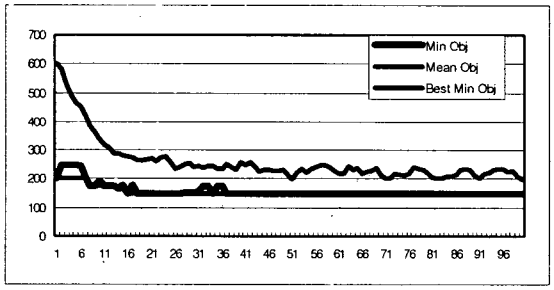
When forward scheduling is used instead of an LP solver, GA computation time is saved at the cost of solution quality. The tests were run 50×100×100 times on a PentiumIII PC using C++.

The parameters used in GA are as follows:

Due date tightness: 1.000000
Large Iteration: 50
Generation No: 100
Population Size: 100
Crossover: 20
Mutation: 2

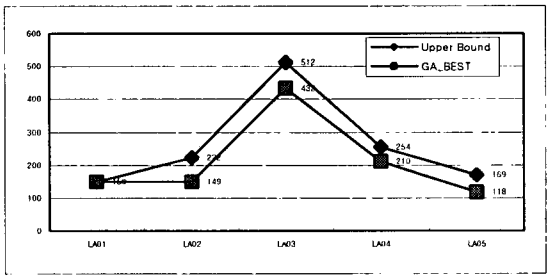
We generate fifty random ROLs and then run GA for each ROL. During GA, 100 initial solutions are obtained which proceed for 100 generations. A policy that conserves the best chromosome over generations is adopted. Experiments show that the mean objective value for 100 chromosomes is dramatically reduced during earlier generations as shown in Figure 3 for LA01.

Figure 3. Progress of GA for LA01



LA01~LA05 are 5×10 JSSP. Although they are relatively easy problems, the optimal solutions for them cannot be obtained in 100 hours. Figure 4 shows the upper bound solved by ILOG CPLEX 7.0. Compared to the optimal procedure, GA generates the lower objective value than that of the MIP optimizer after fifty iterations.

Figure 4. Comparison of best GA solution with optimal upper bound



The dispatching rule ODD (Operation Due Date) recently proposed by Reeja and Rajendran (2000a,b) reported to show good performance in delay minimization. To verify the GA performance, we compare GA to ODD and other dispatching rules such as MWKR (Most Work Remaining) and JDD (Job Due Date). All dispatching rules calculate the solution in 0.5 sec, but they display a wide variation in solution performance as shown in Table 2. In all problems, GA is the best. The value of (dispatching rules' delay / GA delay) reaches a maximum value of 51.29. In particular, JDD generates the worst solution in 33 of the 40 problems. Among the dispatching rules, ODD generates the best solution in 21 problems and MWKR in 17.

Table 1 Comparison of GA with dispatching rules for total tardiness

Problem	ODD	MWKR	JDD	GA_BEST
LA01	575	522	727	150
LA02	918	572	811	149
LA03	1203	847	735	432
LA04	506	515	849	210
LA05	281	344	712	118
LA06	852	902	1329	678
LA07	1562	1434	1729	988
LA08	455	656	897	287
LA09	483	407	1586	345
LA10	1376	1463	1677	577
LA11	322	457	547	201
LA12	358	740	741	147
LA13	635	771	890	424
LA14	536	358	876	155
LA15	524	387	788	256
LA16	18	106	59	0
LA17	291	95	381	50
LA18	44	0	146	0
LA19	662	561	171	54
LA20	217	400	537	0
LA21	335	225	518	0
LA22	178	44	579	0
LA23	197	336	1396	28
LA24	173	462	1080	114
LA25	622	817	887	538
LA26	2647	2995	4696	2405
LA27	3720	3347	4280	2839
LA28	2254	2154	3147	1892
LA29	2772	2204	3421	1105
LA30	2343	1639	3077	1019
LA31	4635	5489	6563	4085
LA32	5696	5995	7503	4285
LA33	4585	4662	7574	4038
LA34	5030	4766	6874	4737
LA35	5068	6344	5807	4453
LA36	0	0	322	0
LA37	336	272	460	196
LA38	1049	1014	648	547
LA39	61	149	406	30
LA40	89	148	718	14

5. Conclusion and further research issues

It is evident that there are disparities between the job-shop scheduling theory and shop floor practices. One of these is manufacturing flexibility, which supports various manufacturing alternatives in production, and another is job hierarchy, which describes the gozinto relationships between jobs. In this paper, we have dealt with the flexible job-shop scheduling problem in the processing of multi-level jobs considering complex routings and alternative machines. This paper has proposed a new gene design under the framework of genetic algorithm, to represent machine assignment, operation sequences, and the relative level of the operation to the final operation. The relative operation level is used as the control parameter that synchronizes the completion timing of the components belonging to the same branch in the job hierarchy. We compared the effectiveness of the genetic algorithm utilizing randomly generated relative levels with that of several dispatching rules in terms of delay. The genetic algorithm revealed outstanding performance in the solution of forty modified standard job shop problems and shows good promise as a scheduling tool in an MRP environment.

References

Anwar, M.F. and Nagi, R. (1997), Integrated lot-sizing and scheduling for just-in-time production of complex assemblies with finite set-ups, International Journal of Production Research, 35(5), 1447-1470.

Brandimarte, P. (1993), Routing and scheduling in a flexible job shop by tabu search, Annals of Operations Research, 22, 158-183.

Kimms, A. (1999), A genetic algorithm for multi-level, multi-machine lot sizing and scheduling, Computers and Operations Research, 26, 829-848.

Lawrence, S. (1984), Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques (Supplement), Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania, USA.

Lourenço, H.R. (1995), Job-shop scheduling:

Computational study of local search and large-step optimization methods, European Journal of Operational Research, 83, 347-364.

Mastrolilli, M. and Gambardella, L.M. (2000), Effective neighbourhood functions for the flexible job shop problem, Journal of Scheduling, 3, 3-20.

Park, M.-W. and Kim, Y.-D. (2000), A branch and bound algorithm for a production scheduling problem in an assembly system under due date constraints, European Journal of Operational Research, 123, 504-518.

Reeja, M.K. and Rajendran, C. (2000), Dispatching rules for scheduling in assembly jobshops - Part 1, International Journal of Production Research, 38(9), 2051-2066.

Reeja, M.K. and Rajendran, C. (2000), Dispatching rules for scheduling in assembly jobshops - Part 1, International Journal of Production Research, 38(10), 2349-2360.

Roach, A. and Nagi, R. (1996), A hybrid GA-SA algorithm for just-in-time scheduling of multi-level assemblies, Computers and Industrial Engineering, 30(4), 1047-1060.