Lecture Notes in Management Science (2013) Vol. **5**: 133–142
5th International Conference on Applied Operational Research, Proceedings

ISSN 2008-0050 (Print), ISSN 1927-0097 (Online)

A hybrid GA for simultaneously scheduling an FMC under multiple objectives

R. Tavakkoli-Moghaddam [1], M. Heydar [2], and S.M. Mousavi [3]

[1] *School of Industrial Engineering,*
South Tehran Branch, Islamic Azad University, Tehran, Iran
tavakoli@ut.ac.ir

[2] *Department of Industrial and Manufacturing Engineering,*
University of Wisconsin, Milwaukee, USA
mheydar@uwm.edu

[3] *Young Researches Club,*
South Tehran Branch, Islamic Azad University, Tehran, Iran

Abstract. This paper solved a bi-objective scheduling problem in flexible manufacturing cell. The objectives considered are maximum completion time (makespan) and maximum tardiness. This class of scheduling problem, regardless of the criterion, belongs to the class of NP-hard problems. Therefore, exact methods are not able to solve practical cases of these types of problems. For this research, a new hybrid genetic algorithm (HGA) combined with four priority dispatching rules is proposed. For numerical study purposes different scheduling problems are generated and solved using the proposed HGA. The results show that the proposed approach performs well in terms of efficiency and quality of the solutions.

Keywords: hybrid genetic algorithm; flexible manufacturing cell; makespan; tardiness

Introduction

Flexible manufacturing system (FMS) aims to achieve the flexibility of low volume production while retaining the efficiency of high-volume mass production. To achieve the efficiency, different decisions can be considered, such as the selection of machines, assignment of the operations and required tools to machines. For the scheduling problems, a common set of resources, including labor, material, and equipment, should be employed to make various products during a given period of time (Srinoi et al., 2006). Complexity of FMS scheduling problems is greater than in classical scheduling problems, and mathematical programming approaches need to be better suited and improved for real-world FMS scheduling problems (Liu and McCarthy, 1998; Kim, 1990; McCarthy and Liu, 1993; Jerald et al., 2005; Snakar et al., 2005). Therefore, the success of an FMS lies in the design of an appropriate scheduling procedure which optimizes the performance measures of such a system.

Scheduling of flexible manufacturing systems received enormous attention over the last three decades. Low and Wu (2001) developed a 0-1 programming for scheduling part in flexible cell. The objective function is minimization total tardiness and jobs are subject to set up times. Then, they developed a heuristic method to solve the problem in a reasonable time. Choi and Lee (2004) proposed a mixed integer programming model for job sequencing in order to minimize the makespan. Chan and Chan (2001) conducted a simulation modeling study on a flexible manufacturing system which minimizes three performance criteria simultaneously, i.e. mean flow time, mean tardiness and mean earliness. Noorul Haq et al. (2003) proposed a multi level scheduling for FMS to generate realistic schedules for the efficient operation of the FMS. Kim et al. (2004) introduced a new GA called network-based genetic algorithm for scheduling jobs in FMSs. Jerald et al. (2005) considered two major resources in FMS, i.e. machine and AGV, and developed a genetic algorithm called adaptive genetic algorithm. Srinoi et al. (2006) focused on scheduling in FMSs using a fuzzy logic (FL) approach. Four fuzzy input variables: machine allocated processing time, machine priority, due date priority and setup time priority are defined. The job priority can be regarded as the output fuzzy variable, illustrating the priority status of a job to be chosen for next operation on a machine. Joseph and Sridharan (2011) assessed the routing flexibility of a FMS by using simulation modeling and analysis. The flexibility levels were then ranked based on the routing flexibility measure for the system. Also, the final ranking was validated using FL approach.

This paper takes the scheduling problem into account by considering one special configuration of FMS known as flexible manufacturing cell (FMC). An FMC includes a set of single flexible machines (SFM) and only one material handling device that can be used when it is idle (Liu and McCarthy, 1998), and the whole system is under computer control. In the literature, machine and vehicle scheduling as two independent problems have been addressed, and only on single objective optimization has been focused as the common approach. However, this paper concentrates on both machine and AGV. FMCs are common place within numerous manufacturing companies, offering numerous advantages, such as the production of a wide range of part types with short lead times, low work-in-progress, economical production of small batches and high resource utilization. To handle the complexity in this class, a new hybrid genetic algorithm (HGA) is developed which is hybridized with four priority dispatching rules.

Problem definition and assumptions

Scheduling of the material handling system in FMS has equal importance as of machines and is to be considered together for the actual evaluation of cycle times and due date related criteria such as jobs tardiness. As such, in this paper both completion time and due date related criteria namely makespan and maximum tardiness are considered.

Model assumptions

In this section, assumptions, based on which under-consideration problem is stated and solved, will be presented. In what follows these assumptions are outlined: (1) Processing time of each operation is known in advance; (2) Transportation times between machines are based on the AGV speed and distance between two different machines; (3) Loading and unloading times are considered in the processing time of each operation; (4) Setup times in this model are sequence-dependent; (5) Machines and AGV breakdown are not accounted for; (6) All machines can process every part and related operations, only if equipped with appropriate tools; (7) Tooling constraints are not considered; (8) Each machine can process only one part at a time; (9) Preemption is not allowed; (10) Processing times are scheduling-independent but machine-dependent, i.e. machine eligibility is taken into account; (11) Technological constraints are known a priory; (12) There are two buffers before and after each machine with limited capacity; (13) To avoid system dead lock, it is assumed that there is a central buffer with unlimited capacity to keep in-line parts; (14) As mentioned above, processing times are machine dependent, mathematically, if p_{ij} is processing time of operation j of job i and v_m is speed of machine m to process the assigned job, then $p_{ijm}(= p_{ij}/v_m)$ will be the time needed to process operation j of job i on machine m.

Hybrid genetic algorithm

In this part the proposed HGA is outlined. Genetic algorithms are non-deterministic stochastic search methods that utilize the theories of evolution and natural selection to solve a problem within a complex solution space, or more specifically combinatorial optimization problems (Sakawa, 2001). The element and mechanism of genetic algorithm are representation, population, evaluation, selection, operator and parameter. The proposed algorithm combines some priority dispatching rules in order to generate schedules from just generated chromosomes for evaluation. The elements of the proposed GA are explained hereafter.

Representation: Every solution of the problem has equivalent representation in GA domain. To link each solution to a chromosome, a coding scheme is needed. In this paper each solution is coded as string of integer numbers (Reddy and Rao, 2005), which is called pheno style (Snakar et al., 2005). Here the initial population is randomly generated, so care must be taken in generating feasible solution that maintains the precedence relations of operations related to the same job. This is crucial in job shop-based scheduling. The following example illustrates how this scheme works.

Example: A scheduling situation with 3 work centers and 4 work pieces is considered. There are 11 operations and the chromosomes consist of 14 genes.

J	*1*		*2*				*3*		*4*		
O.	1	2	1	2	3	4	1	2	1	2	3
M.	1	3	2	3	1	2	2	1	3	1	2
Ge.	1	2	3	4	5	6	7	8	9	10	11

Based on the representation above, a sample feasible chromosome will be:

1 3 2 7 4 9 5 6 8 10 11

Fitness Function: Each individual generated is evaluated for its makespan and maximum tardiness. if completion time of job *i* is defined as:

$$C_i = \sum_j O_{ij} \tag{1}$$

And its respective due date is d_i, then its tardiness will be:

$$T_i = \max(0, d_i - C_i) \tag{2}$$

The maximum tardiness, then, is the maximum of absolute deviation of jobs from their due dates. Mathematically, maximum tardiness is:

$$T_{\max} = \max T_i \tag{3}$$

The maximum completion time or makespan will be defined as follows:

$$C_{\max} = \max(C_1, \ldots, C_n) \tag{4}$$

These two criteria or objectives are combined into one to form the objective of the scheduling problem. This combination as mentioned earlier will form the weighted-sum objective function as follows:

$$z = w_c C_{\max}(s) + w_t T_{\max}(S) \tag{5}$$

where $w_c + w_t = 1$. In this paper both objectives have equal importance, i.e. $w_c = w_t = 0.5$ .The goal is to minimize combined objective *z* via genetic algorithm. Since the main goal of multi-objective scheduling is to find those schedules that belong to Pareto set, this weighted-sum fitness function can be applied. It is worth noting that the approach used here is fixed weighed-sum, meaning that the objective weights remain unchanged throughout the genetic search. Another aspect of GA is operators that play a major role in finding (near-) optimal solution. There are three operators: reproduction or selection, crossover and mutation.

Crossover: The technique used here to cross over two chromosomes is named job-based crossover which never violates precedence relations between operations (Reddy and Rao, 2005). Based on this scheme, once two chromosomes are selected as parents, a job is randomly selected and its corresponding operations are directly copied into respective positions of offspring. This method guarantees that precedence relations are not violated. Then, the remaining unfilled positions are fulfilled with operations of another parent.

Example: Chromosomes selected for crossover are:

P1:	1	5	8	2	6	9	3	7	10	4	11
P2:	5	8	1	9	2	3	6	10	7	11	4

Let the job selected be 2 and the corresponding operations of job 2 are 5, 6 and 7.

P1:	1	5	8	2	6	9	3	7	10	4	11
P2:	5	8	1	9	2	3	6	10	7	11	4

Exchanging the operations of job 2 will result in the following offspring:

O1:	5	1	8	2	9	3	6	10	7	4	11
O2:	8	5	1	9	6	2	3	7	10	11	4

Mutation: Operation swap mutation is used. Two random positions on the chromosome are chosen and the operations associated with these positions are swapped. Operation swap mutation may cause infeasibilities in terms of the precedence relations and a repair function is used to eliminate any such infeasibility (Reddy and Rao, 2005).

Repair function: A repair function is used to see that the chromosomes do not violate the precedence constraints (Ulsoy et al., 1997). The four-step procedure below outlines the repair function in details:

Step 1: find positions of the operations that violate the precedence relations;
Step 2: compute the distance between violating operations;
Step 3: If the distance between them is less than half the chromosome length then swap the operations, else go to Step 4;
Step 4: Randomly pick any one operation and insert it before or after the other depending on the precedence.

Selection: The method used here is known as roulette wheel approach that commonly used in practice (Gen and Cheng, 1997). It belongs to the fitness-proportional selection and can select a new population with respect to the probability distribution based on fitness values, i.e. the more fitted a chromosome is, the more chance it has to be selected. Here, it is required to adapt the selection method in such a way that the minimum objective function receives higher chance to be selected. For this purpose a normalization technique based on a scaling method is used. For minimization problem, this normalization is:

$$f'_k = \frac{f_{\max} - f_k + \alpha}{f_{\max} - f_{\min} + \alpha} \tag{6}$$

where f'_k is normalized fitness function, and $f_{\max}$ and $f_{\min}$ are maximum and minimum values of fitness function of chromosomes in the current population. The just produced fitness values for current generation are used to select chromosomes based on roulette wheel approach. This approach gives to each chromosome in the population an opportunity proportionate to its fitness. This probability is calculated as follows:

$$R(k) = \frac{f'_k}{\sum_{K \in pop} f'_k} \tag{7}$$

Population and parameters: The initial population is randomly generated. The number of chromosomes in each generation, crossover and mutation rates, number of generation that algorithm should run to give a satisfying solution are considered as GA parameters that must be initialized at the beginning of GA run.

Termination criteria: Since each heuristic method does not guarantee an optimum value for the problem it is used, an approach is needed to terminate that heuristic. There different methods to terminated a heuristic method for optimization problem. One that is used here is the number of iteration the algorithm is run.

Schedule generator

Dispatching algorithms are widely used for scheduling in industrial practice. The algorithms are based on various dispatching rules that prioritise the products for assignment to machines and AGVs. So, in this paper dispatching rules are incorporated to GA to schedules jobs on machines and AGV. The proposed GA is the hybrid one that incorporates priority dispatching rules to do the scheduling task. To keep track of scheduling in terms of tardiness objective, job slack (Pinedo, 2005) is calculated which is defined as follows:

$$MS = \max\left(d_j - p_{ijm} - t, 0\right) \tag{8}$$

This rule is not considered as those imbedded into GA. The main purpose is to schedule machines in such a way that once a machine becomes free, this index is calculated and based on it the next operation in the current chromosome is scheduled. Four priority rules are earliest completion (finishing) time (EFT), shortest processing time (SPT), shortest distance time (SDT) and fewest waiting jobs for machine (FWJM). The first two focus on jobs, the third one tries to handle AGV constraint and make use of its availability and its impact on the objective functions. The last one plays role the same as previous one except it considers machines buffers.

This proposed methodology work as follows: first, a job with earliest finishing time is selected to be processed on the corresponding machine. If there is more than one job, the job with shortest processing time for its subsequent operation is selected. Then tie is broken by considering the distance each job should travel, i.e. the shortest path is selected first by AGV. If again there is a tie, another PDR is taken into account. Based on this rule, the number of jobs in the target machine buffer determines which job should go first. This GA in conjunction with proposed heuristic approach constructs the methodology presented for scheduling jobs and AGV in a flexible manufacturing cell.

Proposed HGA steps for scheduling FMC

In this section, steps for scheduling a flexible manufacturing cell are presented.

Step 1: Enter input data including number of machines, distance between machines, number of jobs and corresponding operations, processing and setup times and due dates. Enter GA parameters such as population size, crossover and mutation rates and termination criteria.
Step 2: Randomly generate an initial population using the encoding scheme.
Step 3: Generate schedules using schedule-generator module.
Step 4: Using roulette wheel approach select chromosomes to create mating pool for next generation.
Step 5: Generate offspring population using job-based crossover and bit-wise exchange mutation operators. If some precedence relations are violated, go to step 6; otherwise go to step 7.
Step 6: In case of any violation as a mutation result, run repair function as described above and go to step 7.
Step 7: Evaluate each chromosome in current population for objective function based on the generated schedule.
Step 8: Sort chromosomes based on the fitness function value.
Step 9: If termination criterion is satisfied, then stop and print the fittest chromosome as the best solution found; otherwise go to step 4 for next generation.

The next section presents the result of the proposed approach to deal with scheduling problem in a flexible manufacturing cell environment.

Numerical examples and comparison results

In this part, the proposed approach is applied to schedule FMC with varying parameters. The proposed algorithm is coded in Visual C++ 6. Many problems with different parameters and values were considered and solved. The results are tabulated. Since the problem environment is somehow similar to the one considered by Liu and McCarthy (1998), the MILP was solved by Lingo 8 and the results were compared with those of our heuristic approach.

First, ten problem examples were randomly generated and shown in Table 1. In each problem example, different job sets with different operations were considered. Based on the mathematical formulation number of variables and constraints are also calculated and provided in Tables 1 to 3. To further study the efficiency of proposed model, different problems with different configurations are defined in three stages, based on which both mathematical model and GA methodology are applied.

First, an FMC with two machines is considered and iteratively problem size is increased by adding job with varying operations. Processing times are randomly generated from a uniform distribution function, accordingly jobs due dates are defined (Table 2). Then, the configurations of FMC, in terms of distance between machines or case 2 (Table 3) were changed. GA parameters were remained unchanged, though their impacts on algorithm performance can be effective.

Table 1. Data for experimental study

Prob. no	*Jobs*	*Oper. Per job*	*Total Oper.*	*Machines*	*No. of Var.*	*No. of Cons.*
1	4	2	8	2	208	596
2	6	2	12	2	632	985
3	10	3	30	2	3732	4466
4	6	6	36	2	5336	4561
5	7	6	42	2	7233	6308
6	15	3	45	2	8297	10186
7	9	6	54	2	11891	19648
8	10	6	60	2	14652	13241
9	15	6	90	2	32777	30436
10	20	6	120	2	58102	54681

Table 2. Results for case 1

Prob. no	*Global Solution*					*GA Solution*			
	Optimal	*Cmax*	*Tmax*	*Iter.*	*Time(second)*	*BOF*	*Cmax*	*Tmax*	*Time(second)*
1	51.75	103.5	0	8734	8	54.75	108.7	1	30
2	59.9	105.5	14.4	23075	17	64.7	114	15.4	67
3	30.5	48	13	381173	197	31.65	49.4	13.4	150
4	9	18	0	965846	623	9.25	18.5	0	400
5	11.5	23	0	2800953	1987	12	24	0	964
6	27.5	50	5	3591046	3674	29.35	53.5	5.2	1630
7	109.5	190	29	7541197	6743	117.5	205	31	3180
8	162	278	46	11311795	10863	173	297	49	5342
9	204.5	359	50	18664462	15642	218.5	384	53	8334
10	230.5	406	55	35462777	24651	246	434	58	10045

Table 3. Results for case 2

Prob. No	*Global Solution*					*GA Solution*			
	Optimal	*Cmax*	*Tmax*	*Iter.*	*Time(second)*	*BOF*	*Cmax*	*Tmax*	*Time (second)*
1	105.35	157.8	52.9	17234	10	112.75	170	55.5	45
2	152.65	197.8	107.5	380560	110	160.4	208	112.8	150
3	35.5	54	17	1376544	615	37	56	18	700
4	11.5	23	0	7764906	3185	13	25	1	1285
5	35.6	57	14	14753321	7348	38	61	15	2059
6	64.5	109	20	32457307	15325	69	117	21	3942
7	177	300	54	63291749	29841	189	321	57	7002
8	-	-	-	-	-	230	400	60	9358
9	-	-	-	-	-	262.5	463	62	11803
10	-	-	-	-	-	289	513	65	14391

This algorithm is coded in Visual C++ 6 along with coded MILP model in Lingo 8. Both were run on a PC with 2.6 GHz CPU and the results are tabulated in the following page. It can be concluded that how the configuration and layout of manufacturing cell can increase the problem complexity. As results show, using the GA to solve this problem will reduce time needed to get best objective function dramatically showing that using this technique is promising. Another fact that is worth mentioning is the impact that FMS layout has on production planning in general and scheduling in particular, i.e. the more machines were located away, the greater the completion time, and also tardiness, is. For this reason layout of a cell must be considered in scheduling system design.

Conclusion

Flexibility is a growing issue in modern industrial firms to respond varying product demand with short lifecycle. Therefore, new approaches are needed to resolve this issue. Since FMS scheduling problems are NP-hard, using heuristic methods are quite justified. In this paper a class of FMS known as flexible manufacturing cell is considered. A new HGA-based approach is proposed to schedule jobs and AGV for minimizing makespan and maximum tardiness, simultaneously. The hybrid algorithm is coded in Visual C++ and run for problems of different sizes. One reason that is worth considering is the required time to solve medium to large size problems that is a crucial issue in industrial firms. As new direction for future research, the current study can be further extended by applying other heuristic methods separately or in conjunction with HGA algorithm.

References

Chan FTS, and Chan HK, (2001). Dynamic scheduling for a flexible manufacturing system: the pre-emptive approach. International Journal of Advanced Manufacturing Technology, 17: 760-768.

Choi SH, and Lee JSL, (2004). A sequencing algorithm for makespan minimization in FMS. Journal of Manufacturing Technology Management, 15(3): 291-297.

Gen M, and Cheng R, (1997). Genetic algorithms and engineering design. John Wiley & Sons.

Jerald JP, Asokan R, and Rani ADC, (2005). Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithm. International Journal of Advanced Manufacturing Technology, 29(5): 548-589.

Joseph OA, Sridharan R, (2011). Evaluation of routing flexibility of a flexible manufacturing system using simulation modelling and analysis, International Journal of Advanced Manufacturing Technology, 56(1-4): 273-289.

Kim KW, Yamazaki G, Lin L, and Gen M, (2004). Network-based hybrid genetic algorithm for scheduling in FMS environment. Artificial Life and Robotics, 8: 67-76.

Kim YD, (1990). A comparison of dispatching rules for job shops with multiple identical jobs and alternative routings. International Journal of Production Research, 28(5): 953-962.

Liu J, and McCarthy BL, (1998). A global MILP model for FMS scheduling. European Journal of Operational Research, 100: 441-453.

Low C, and Wu TH, (2001). Mathematical modeling and heuristic approaches to operation scheduling problems in an FMS environment. International Journal of Production Research, 39(4): 689-708.

McCarthy BL, and Liu J, (1993). A new classification for flexible manufacturing systems. International Journal of Production Research, 31(2): 299-309.

Noorul Haq A, Karthikeyan T, Dinesh M, (2003). Scheduling decision in FMS using a heuristic approach. International Journal of Advanced Manufacturing Technology, 22(5-6): 374-379.

Pinedo ML, (2005). Planning and scheduling in manufacturing and services. Springer.

Reddy BSP, and Rao CSP, (2005). A hybrid multi-objective GA for simultaneous scheduling of machines and AGVs in FMS. International Journal of Advanced Manufacturing Technology, 31(5-6): 602-613.

Sakawa M, (2001). Genetic algorithms and fuzzy multi-objective optimization. Kluwer Academic Publisher.

Snakar SS, Ponnambalam SG, and Rajendran C, (2005). A multi-objective genetic algorithm for scheduling a flexible manufacturing system. International Journal of Advanced Manufacturing Technology, 22 (3-4): 229-236.

Srinoi P, Shayan E, Ghotb F, (2006). A fuzzy logic modelling of dynamic scheduling in FMS, International Journal of Production Research, 44(11): 2183-2203.

Ulsoy G, Serifoglu FS, and Bilge U, (1997). A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. Computers and Operations Research, 24 (4): 335—351.