Scheduling Problem of Virtual Cellular Manufacturing Systems (VCMS); Using Simulated Annealing and Genetic Algorithm based Heuristics

Saeed Taouji Hassanpour[1*], Reza Bashirzadeh[2], Abolfazl Adressi[3], Behnam Bahmankhah[4]
[1] Phd Candidate at Tarbiat Modares University, Tehran, Iran
[*]Email of Corresponding Author: Saeed.tasouji@modares.ac.ir
[2] Assistant Professor at K. N Toosi University of Technology, Tehran, Iran
[3] Master of Industrial Engineering at K. N Toosi University of Technology, Tehran, Iran
[4] Phd Candidate at University of Aveiro, Aveiro, Portugal
Received: May 30, 2015; Accepted: June 21, 2015

Abstract
In this paper, we present a simulated annealing (SA) and a genetic algorithm (GA) based on heuristics for scheduling problem of jobs in virtual cellular manufacturing systems. A virtual manufacturing cell (VMC) is a group of resources that is dedicated to the manufacturing of a part family. Although this grouping is not reflected in the physical structure of the manufacturing system, but machines are spread on the shop floor physically. In this paper, there are multiple jobs with different manufacturing processing routes. First, we develop the mathematical model for the problem, and then we present the suggested algorithms. The scheduling objective is weighed tardiness and total travelling distance minimization. The problem is divided into two branches: small scale and large scale. For small scale, the results of GA and SA are compared to GAMS. For large scale problems, due to the time limitation of 3600 seconds, the results of GA and SA are compared to each other. Computational results show that both SA ad GA algorithms perform properly but SA is likely to turn out well in finding better solutions in shorter times especially in large scale problems.

Keywords
Virtual cellular manufacturing systems, Scheduling, Simulated annealing, Genetic algorithm, mathematical formulation

1. Introduction

A system should be able to respond to product designing changes and production demands without requiring great deals of investments. It should necessarily conform itself with the new conditions. Manufacturing systems should continuously accommodate themselves with the dynamic conditions of the global market so that they can survive in the competitive world of business and manufacturing. Traditional production methods such as product type and functional type are not able to provide these requirements properly, therefore a new method is required to obtain these vital characteristics. Group technology (GT), a strategy suggested for these new requirements, is a manufacturing method through which the parts that have similarities, named part families, are grouped together in order to achieve the said goals. The concept of the group technology has resulted in the cellular manufacturing (CM) and virtual cellular manufacturing (VCM) systems. These two manufacturing methods have attracted a lot of attention during recent years.

Prior studies have shown that cellular manufacturing is superior to job shop manufacturing in cases which require long set-up and long material handling times, and where customer demand is stable (Greene and Sadowski [1], Wemmerlov and Hyer [2], Morris and Tersine [3]). The virtual cellular manufacturing system has its roots in the concept introduced by Mclean et al [4]. Virtual cellular manufacturing system (VCMS) is a branch of CM in which the cells are not physically defined and can be changed during the manufacturing process according to the production schedule. A virtual cell, deduced from the word virtual, is a kind of cell that possesses virtual character, meaning, in a period of time it exists but in other period can be eliminated. In fact the cells are not considered as a fixed physical grouping of machines spread on the floor shop but as data files in a virtual cell controller. The difference between traditional cells and virtual cells is that the assignments of the jobs in a virtual manufacturing cell are altered by a controller named virtual cell controller periodically. The cells are not physically fixed in virtual cells and they can change without displacing the machines. The workstations in a virtual system will not be incarcerated on the formation of a virtual manufacturing cell, but on taking advantage of the machine capacities they can be assigned to other cells to perform operations provided that there are excess capacities. Similar to the traditional cells, some promising advantages of virtual cells are better quality and production control. Other advantages of this approach include higher efficiency, more improved flow performance and better flexibility than the CM.

Virtual concept of VCM, to some extent, has unraveled some problems of CM such as machine utilization and unbalanced workload. Also great improvements are detectable in the results, but many aspects of VCM such as scheduling of VCM have not attracted a great deal of attention from researches due to the complexity of the model.

Scheduling problem of VCMS is similar to the scheduling problem of Jobshop with two significant differences. The distinction between scheduling problem of VCMS and Jobshop lies in machine type and total travelling distance. The concept of machine type existing in VCMS means that there are multiple machines with same characteristics of each machine type and are spread on the shop floor. Provided that each machine type is regarded as a single machine, the problem will be propelled to Jobshop problem. The second distinction between these two issues is that total travelling distance should be taken into consideration as one of the objective functions. It is more realistic that a process which needs two machine types to complete its route is processed on two machines that are close to each other than two machines with notable distance. In this case the concept of forming different cells will become much more tangible. Thus, scheduling of VCMS has more complexity in comparison with Jobshop because it requires the machine selection among some identical machines of a machine type in addition to sequencing of operations.

2. Literature Review

Slomp et al [5] designed virtual manufacturing cells with mathematical programming. Kesen et al. [6] defined three types of systems as follow: cellular layout (CL), process layout (PL) and virtual cells (VC). They used a simulation approach to compare these methods under performance metrics such as mean flow time and mean tardiness. They also proposed an ant colony optimization model based upon the available simulations to represent the future simulations. The results showed that VCs are more flexible compared to other mentioned methods without taking into account the setup time. Kesen et al. [7] presented a genetic algorithm based approach for scheduling of VCMS

considering the weighed makespan and total traveling distance as the objectives for minimization of the objective function. The results of the genetic approach were compared with the mixed integer programming (MIP) approach [8]. The results show that genetic algorithm can be easily substituted with the MIP model. Mak et al [9] developed a new mathematical method for formulation of the scheduling of VCMS. They divided the total scheduling horizon into several time periods during which the product mix and demand are not the same, but they are deterministic. These researches assumed that when first operation of any job starts, next operation must start in the next period. They also assumed that successive operation would only start at the beginning of the next period even if the preceding one were finished earlier. They utilized constraint programming to deal with the intricate nature of the problem. Mak et al. [10] presented a hybrid approach based on discrete practical swarm optimization and constraint programming to solve the scheduling problem of VCMS. The objective taken into account was total travelling distance over the entire planning horizon. The results revealed that the hybrid method, especially for large scale problems, can result in better scheduling solutions. Babu et al [11] for SMEs developed virtual cellular manufacturing system. Khilwani et al [12] designed a new methodology for virtual cellular manufacturing systems. Mahdavi et al [13] consider multi objective cell formation and production planning in dynamic VCMs. Kannan [14] considers a simulation analysis of the impact of family configuration for VCMs. Nikoofarid and Aalaei [15] consider production planning and worker assignment in a dynamic cellular manufacturing systems.Hamedi et al [16] solve capability-based virtual cellular manufacturing systems formation in dual-resource constrained settings by using Tabu search algorithm. The remaining sections of this paper are organized as follows:
The problem description and mathematical model is presented in section 3. In sections 4 and 5, we introduce the genetic algorithm and simulated annealing based heuristic respectively. Conclusion and Computational results are presented in section 6 and finally references are in section 7.

3. Problem description and Mathematical model

In this section, a mathematical model is presented to describe the characteristics of the proposed model. Before developing a mathematical model, the following assumptions are made: In the problem there are *n* jobs and *m* machine types. The scheduling objective is weighed tardiness and total travelling distance minimization. Each machine type is able to perform a special operation and consists of several identical machines located in different locations on the shop floor. All machines of a machine type have the same speed and characteristics. Each job has a pre-determined processing route and should get through machine types according to the scheduled program, that is, the machines by which each operation of a job can be performed are known in advance. Each operation of a job has a specified processing time on the related machine types which is independent of the job's processing route and processing order. Each operation is processed only on one of the machine types and includes tardiness penalty per each tardiness unit and delivery time (due date). $o_{i,j}$ represents the *j*th operation of the job *i*. If $o_{i,j}$ is performed on machine type *k*, all of the individual machines of that type can be chosen, so there is a competition among these machines and this makes the problem expose to two issues: machine assignment for the operations and scheduling of operations. Each job can visit machine type *k* at most once. Jobs are presented in batches. Some information about the jobs such as batch sizes, processing times, operation sequence,

transportation cost of each job and distance between each pair of machines are known. When $o_{i,j}$ is assigned to a machine type *k*, all of the batch operations must be performed on that machine. Interruption during the processing is not allowed.

3.1 Assumptions

No preemption is allowed in the model. All jobs are available at zero time. We are not allowed to move the machines. The transportation time and the machine setup time can be disregarded. Breakdown and maintenance time and costs are not considered in the model. The parameters and variables used in this model are presented as follows:

Indexes:

i	job index (i=1,...,n)
j	operation index (j=1,...,m)
k, k'	machine group type (k,k'=1,...,m)
s, s'	machine index belonging to a specified group (s,s'=1,...,s_k)
l	order index for each machine (l =1, 2, … , l_k)

Parameters:

$o_{i,j}$	jth operation of the job i
p_{ijk}	processing time of $o_{i,j}$ on machine type k
a_{ijk}	1 if $o_{i,j}$ is performed on machine type k, 0 otherwise
s_k	number of machine type k
w_i	Tardiness penalty of job i
dd_i	due date of job i
WF_h	weight of hth objective function
$D_{ksk's'}$	distance between machine s of group k and machine s' of group k'
C_i	unit transportation cost for job i
N_i	batch size of job i
M	very big number which can be considered as sum of processing times of all the operations

Variables:

s_{ij}	starting time of the $o_{i,j}$
sm_{ksl}	starting time of the operation positioned on sth machine of machine type k for the order l

y_{ijk}^{s}	1 if $o_{i,j}$ which needs to be operated on machine type k is performed on sth machine of type k, 0 otherwise
r_{ijk}^{sl}	1 if $o_{i,j}$ which needs to be operated on machine type k is performed on sth machine of type k for order l, 0 otherwise
CT	completion time for job i
T_i	tardiness of job i

3-2-Mathematical model:

$$\text{minimize} \quad WF_1 \sum_i w_i T_i + WF_2 * \sum_i \sum_j \sum_k \sum_s \sum_{k'} \sum_{s'} C_i D_{ksk's'} y_{ijk}^{s} * y_{i,j+1,k'}^{s'} \tag{1}$$

$$\sum_s y_{ijk}^{s} = a_{ijk}; \forall i, j, k \tag{2}$$

$$\sum_l r_{ijk}^{sl} = y_{ijk}^{s}; \forall j, k, s \tag{3}$$

$$\sum_i \sum_j r_{ijk}^{sl} \le 1; \forall k, s, l \tag{4}$$

$$s_{i,j-1} + \sum_k \sum_s \sum_l N_i * p_{i,j-1,k} * r_{i,j-1,k}^{s,l} \le s_{i,j}; \forall i, j > 1 \tag{5}$$

$$sm_{k,s,l-1} + \sum_i \sum_j N_i * p_{ijk} * r_{ijk}^{s,l-1} \le sm_{k,s,l}; \forall k, s, l > 1 \tag{6}$$

$$s_{ij} \le (1 - r_{ijk}^{sl}) * M + sm_{ksl}; \forall i, j, k, s, l \tag{7}$$

$$sm_{ksl} \le (1 - r_{ijk}^{sl}) * M + s_{ij}; \forall i, j, k, s, l \tag{8}$$

$$CT_i \ge s_{ij} + \sum_k \sum_s \sum_l N_i * p_{ijk} * r_{ijk}^{sl}; \forall i, j \tag{9}$$

$$CT_i - dd_i \le T_i; \forall i \tag{10}$$

$$r_{ijk}^{sl} = \{0,1\}, y_{ijk}^{s} = \{0,1\}, s_{ij} \ge 0, sm_{ksl} \ge 0, \quad CT_i \ge 0, T_i \ge 0; \forall i, j, k, s, l \tag{11}$$

Objective function consists of minimizing weighed sum of two objectives: tardiness and total travelling distance. Equation (2) ensures that each operation of each job can be assigned to just one particular machine of the related machine type. Equation (3) denotes that when operation of one job is assigned to any particular machine, this operation can be positioned in any order of the machine. Constraint set (4) guarantees that on each order of any machine of any type, we can assign one machine at most. Constraint set (5) adjusts the starting time of operations which are positioned in the processing route, in other words, it makes sure that successive operations of any machine are performed after the preceding ones. Constraint set (6) is similar to (5) and indicates that operations with successive priority must wait for the completion of preceding ones. Constraints (7) and (8) are

used to adjust the starting time of each operation of each job and starting time of jobs on machines, that is, if r_{ijk}^{sl} is equal to 1, then $o_{i,j}$ and *l*th order of *s*th machine of type *k* must start simultaneously. Constraint (9) restricts the completion time of each job to be equal or greater than the completion times of all the operations of that job. Constraint set (10) shows the earliness and tardiness of each job according to the completion time and due date of that job. Finally, constraint set (11) explains the non-negativity conditions of the variables.

Regarding that the second part of the objective function is non-linear, we defined a new variable named $z_{i,j,k,j+1,k'}^{s,s'}$ to eliminate the non-linearity of the model.

$$z_{i,j,k,j+1,k'}^{s,s'} = y_{ijk}^{s} * y_{i,j+1,k'}^{s'} \tag{12}$$

$$z_{i,j,k,j+1,k'}^{s,s'} - y_{ijk}^{s} - y_{i,j+1,k'}^{s'} + 1 \geq 0; \forall i, j < n_{i,k,s,l} \tag{13}$$

$$2 * z_{i,j,k,j+1,k'}^{s,s'} - y_{ijk}^{s} - y_{i,j+1,k'}^{s'} \leq 0; \forall i, j < n_{i,k,s,l} \tag{14}$$

By applying the changes mentioned above on the objective function, objective function will be varied as follows:

$$\text{minimize} WF_1 \sum_i w_i T_i + WF_2 * \sum_i \sum_j \sum_k \sum_s \sum_{k'} \sum_{s'} C_i * D_{ksk's'} * z_{i,j,k,j+1,k'}^{s,s'} \tag{15}$$

Also, Constraints (13) and (14) must be added to the model.

4. Proposed GA approach for scheduling VCMs

VCM problem usually deals with two different issues: machine assignment and scheduling problem of operations. The representation can be displayed as a kind of expanded job-list, which consists of ***N*** x *M* genes, where ***N*** is the number of jobs and *M* is the number of machines

The objective function value of all chromosomes are calculated and ordered in descending way. Fitness function chromosome *i* is calculated by equation (16) where FF_i , OF_w and OF_i represent fitness function for *i*th chromosome, the worst objective function available and objective function of current chromosome respectively. The equation is added by 1 in order to make it possible for the worst chromosome to be selected for the next population.

$$FF_i = OF_w - OF_i + 1 \tag{16}$$

The evolution process of GA usually starts from a randomly generated population. In each generation, the fitness of each individual chromosome in the population is assessed. The more fitness function of an individual, the more chance of being selected will be. New generation is stochastically selected from the current population, and each individual's genome is modified by means of mutation and crossover operator to form a new population. The new population is then used in the next iteration of the algorithm. The initial population for the proposed GA algorithm is randomly generated.

In this paper, in order to perform a mutation operator, two genes are selected randomly and then their positions are replaced by each other.

There are different types of crossover operators used in GA such as one-point, two-point, uniform and arithmetic ones. In this paper, a hybrid of uniform and two-point crossover have been utilized. A number between 0 and 1 israndomly generated. When its value is more than 0.5, the uniform crossover will be applied and when the value is less than 0.5, the two-point cross over will be considered.

Survival selection procedure is executed after generating the offspring. Size of the population is fixed and will not alter during the procedure, therefore we should determine which individuals are participating in the next generation. This decision is usually taken according to the fitness function values. Solutions in the population are sorted from the best to the worst in accordance with their performance on the objective function of the problem. In this paper for selecting the survivals, three approaches are implemented: crossover operator, mutation operator and elites (chromosomes transferred to next generation without any change). In order to generate better solutions, a local search approach is performed on 50 percent of the new generation.

Different terminations conditions can be applied to a GA algorithm. For the proposed GA algorithm, one of the following conditions causes the algorithm to reach to its end:

1. Generating a specified number of generations.
2. No improvement is observed during a specified period of generations.

5. Proposed SA approach for scheduling VCMs

Representation of solutions is the same as the one used in the GA algorithm. To create a new neighborhood, two genes are selected and interchanged with each other.

Determination of the initial temperature is very important in accepting or rejecting the solutions. The higher the temperature, the more significant the probability of accepting a worst move will be. On the other hand, low degrees of temperature reduce the acceptance probability of bad solutions and increase the chance of remaining in a local optima.

Cooling schedule has a great impact on the success of the SA optimization algorithm. The parameters to be considered in defining a cooling schedule are the initial temperature, the equilibrium state, a cooling function, and the final temperature. Different methods to decrease the temperature degree are arithmetical, linear, geometric, logarithmic, very slow decrease and non-monotonic. In this paper, we will use arithmetical method with the constant value of C equal to 0.8.

$$T_k = T_{k+1} - C \tag{17}$$

To reach an equilibrium state at each temperature, a number of sufficient moves must be applied. This algorithm requires to be speculated in a specified temperature degree after some iterations to make a decision of continuing the annealing process in that degree or terminating the process and stepping to the other degree. In most SA methods, a number of specified replacements are taken place in a temperature degree named epoch or period for assessing the equilibrium conditions. Number of these replacement is shown by N. We have employed following constraint to speculate equilibrium condition:

$$\frac{\left|\bar{f}_e - \bar{f}_e^{'}\right|}{\bar{f}_e^{'}} \leq \varepsilon \tag{18}$$

Where $\bar{f}_e$, $\bar{f}_e^{'}$ and ε stand for objective function average in last epoch for all of the accepted replacements, average of all amounts of $\bar{f}_e$ and errors respectively.

We have considered two termination conditions. The first one is to reach to final temperature degree. The second is to achieve all of the generated neighborhoods or all of the accepted replacements during algorithm running time.

6. Conclusion and Computational results

The developed mathematical model for solving the proposed problem is coded in GAMS/Cplex 22.5. Optimization software and GA and SA algorithms are coded in C++ Borland 6.0 on a computer with 4GB RAM, Intel Core2 Duo P7550 CPU, 2.26 GHz processor. Time limitation for each generated problem is 3600 seconds. In real world 20 percent of orders are of high importance, 60 percent are of mediocre importance and the rest are of low importance orders. Considering this issue, for 20 percent of jobs, the tardiness penalty is considered equal to 4, for 60 percent of jobs is 2 and for the rest of them is 1. Due date tightness function is also regarded as one of the factors in calculating the due date. In job shop literature, values of 1.3, 1.5 and 1.6 are considered for this factor. As this factor gets smaller, the problem becomes harder to be solved.

$$d_i = N_i * f * \sum_{j=1}^{n_i} p_{ij} \tag{19}$$

In equation (19) d_i, N_i, f, p_{ij} and n_i are due date of job *i*, batch size of job *i*, due date tightness factor, process time of *j* th operation of job *i* and the number of operations belongs to job *I* respectively. Process routes of the problem are randomly generated. Processing time of the operation on machines corresponds to uniform distribution with a lower bound 2 and upper bound of 10. Batch size of each job belongs to uniform distribution range between 5 and 40. Distances between each pair of machines also correspond to uniform distribution with lower bound 10 and upper bound 20. Unit transportation costs between each pair of machines follow uniform distribution with a lower bound of one and upper bound of 5. The weight of tardiness and total travelling distance part of the objective function are determined 0.95 and 0.05 respectively. The problem is divided into two branches: small scale and large scale. Small scale includes problems with 4, 6, 8 and 10 jobs. Large scale problems consist of 15, 20, 25 and 30 jobs. As it can be seen in Table 3 and Table 7, four aspects have been brought into account in order to generate different problems: scale of the problem, a number of machine types (M), a number of individual machines that exist in each machine type and due date tightness factor. Considering these four aspects, 288 different problems were generated. Some of them are listed in Table 3 and Table 7. *s1, s2* and *s3* are representatives of number of machines in machine types and stand respectively for {2}, {2,3} and {3,4}. For example, *s2* means that number of machines related to each machine type should be randomly chosen between 2 or 3 machines. On account of non-deterministic nature of GA algorithms, we conducted the GA four times for four types of the proposed problem to gain the

parameters of the algorithms experimentally. By considering these results, the mutation, crossover and reproduction (elite) values for small scale problems are %7, %80 and %13 respectively. For large scale problems, the values are %10, %70 and %20. The values of population size, number of maximal generations and number of local search as can be seen in Table 1 are obtained experimentally too. In order to prevent computational time increasing, local search is applied only on 50 percent of the population.

Parameters of SA are defined in two phases. In phase one, in order to obtain the best combination for ε and N_k, the parameters of Table 2 were considered. In the second phase, through values gained from the first phase for ε and N_k, we defined the Initial temperature, final temperature and Boltzmann constant for the problem. We conducted 17 runs of proposed algorithms to obtain the best combination for ε and N_k. The results of phase one showed that for small scale, the best combination is 0.008 and 3 respectively. For large scale problems, the values were set as 0.003 and 15. The result of phase two determined values of initial temperature, final temperature and Boltzmann constant equal to 50,1 and 1 for small scale problems and 100, 1 and 1 for large scale ones. The results of this test can be seen in Table 3.

We can observe that average solution time increases as the number of workstations intensifies. SA is better than GA in the point of average error and number of optimal solutions especially in large scaled ones. Both GA and SA excel GAMS in the given computational time limitation. Decreased computational time and low value of the errors demonstrate the efficiency of proposed methods.

Table1. Population, generation and local search information

Number of jobs	Population size	Number of Generation	Local search
2,4,6,8	100	100	5
10,15	150	100	7
20	200	150	7
25,30	250	150	7

Table2. Phase one parameters

Initial temperature	100
Constant value of temperature function	0.8
Final temperature	1
Boltzmann constant	1

Table3. Computational results for small scale problems for GAMS, SA and GA

Parameters				GAMS		SA		GA		Gap	
				Solution	Time	Solution	Time	Solution	Time	SA	GA
N=4	*M=3*	*s1*	*f=1.3*	142.85	1.357	142.85	3.8	142.85	1.94	0	0
N=4	*M=3*	*s3*	*f=1.6*	142.4	3.121	143.25	6.2	142.4	2.92	0.0059	0
N=6	*M=3*	*s1*	*f=1.3*	556.45	234	556.45	10.08	556.45	3.84	0	0
N=8	*M=3*	*s2*	*f=1.5*	268.55	5	268.55	12.8	268.55	3.58	0	0
N=10	*M=3*	*s3*	*f=1.6*	375.85	10	386.65	13.9	378.45	5.1	0.0234	0.0069
N=4	*M=5*	*s3*	*f=1.6*	534.75	2.519	534.75	19.59	534.75	6.1	0	0
N=6	*M=5*	*s1*	*f=1.3*	901.65	9.173	906.15	10.42	903.45	6.1	0.0049	0.0019
N=8	*M=5*	*s2*	*f=1.5*	1123	162.21	1123	24.9	1148.3	7.9	0	0.0225
N=10	*M=5*	*s3*	*f=1.6*	1126	177	1158.3	33.44	1179.55	26	0.286	0.0475
N=4	*M=7*	*s3*	*f=1.5*	957.3	7.246	959.1	23.33	984.1	8.98	0.0018	0.0279
N=6	*M=7*	*s1*	*f=1.6*	2088.45	13.496	2088.45	6.62	2090.4	8.44	0	0.0009
N=8	*M=7*	*s2*	*f=1.3*	1462	243	1494.3	36.2	1561.6	12.08	0.0220	0.0681
N=10	*M=7*	*s3*	*f=1.5*	1947.3	481	1954.3	42.34	2076.65	25.21	0.0035	0.0664
N=4	*M=10*	*s3*	*f=1.6*	1182.5	45.458	1194.9	44.37	1217.45	22.15	0.0104	0.0295
N=6	*M=10*	*s1*	*f=1.3*	1287.8	48.173	1289	11.56	1293.2	21.44	0.0009	0.0041
N=8	*M=10*	*s2*	*f=1.3*	4356.05	1657	4475	7.135	4497	29.2	0.0273	0.0323
N=10	*M=10*	*s3*	*f=1.6*	3755.5	3600	3620.8	71.15	3879.25	52.1	0.0358	0.0329

We can see that in small scale problems, objective function values in all problems are almost close to each other. By comparing these solutions with the ones obtained from GAMS software, the proposed model and the results of the proposed algorithms can be properly validated.

Also, it can be seen that in most cases, the time taken to achieve the optimal solution in the GA is less than the SA. This makes the proposed GA to be more effective in solving the small scale problems (except for problems of size 15 and 16). The results of this test can be seen in Figure Figures 1 and 2.

In order to check the equality of the values resulted from the proposed objective function in GA and SA, we use the hypothesis testing. First, using the objective functions values, we checked the normality of them at 95% confidence level according to the results obtained. In small scale problems, the p-value was less than 0.05 which indicates that the results are not normal.

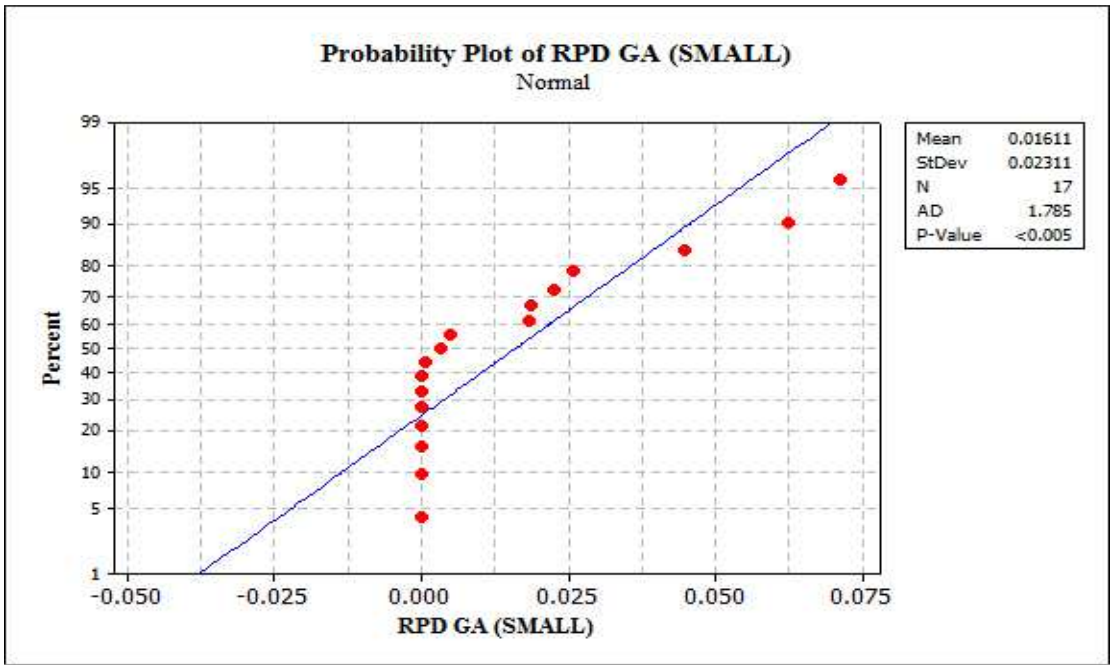


Figure1. Normality test for the proposed GA in small scale problems

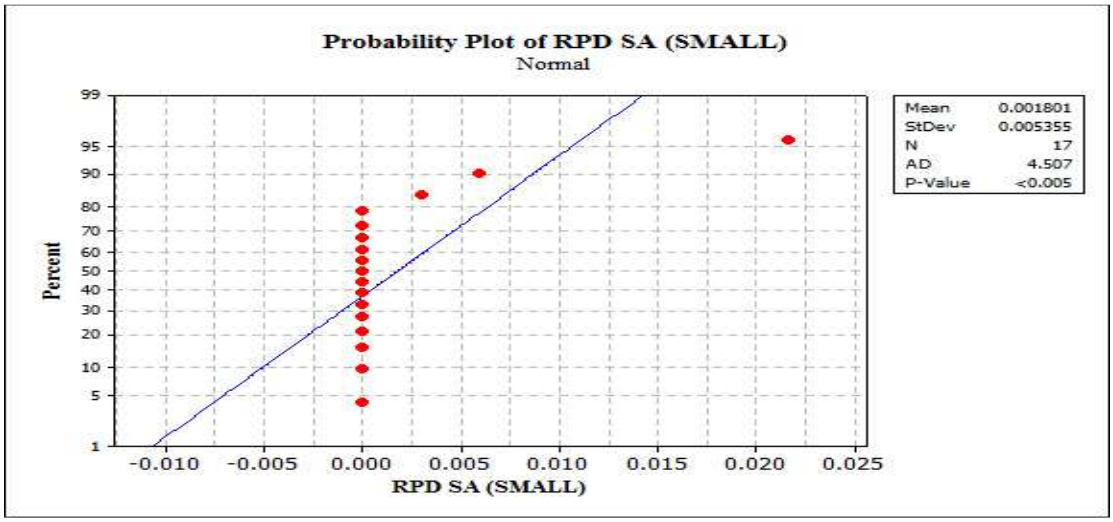
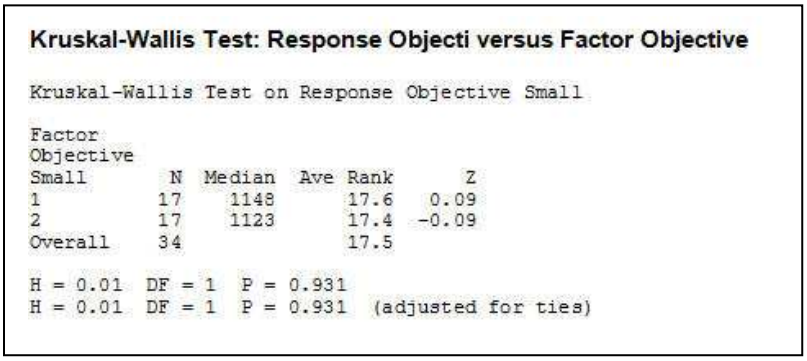


Figure2. Normality test for the proposed SA in small scale problems

For this reason (having non-normality data), we use non-parametric statistical tests to check for equality of means obtained from the use of the proposed algorithms. In this article, Kruskal-Wallis non-parametric test was used. The results of this test can be seen in Figure 3. As it can be seen from Figure 3, as well as the P=0.931, we can conclude that equality of the values obtained from the proposed algorithm cannot be rejected at the 95% confidence level. The results of this test can be seen in Tables 4, 5 and 6.



Kruskal-Wallis Test: Response Objecti versus Factor Objective

```
Kruskal-Wallis Test on Response Objective Small

Factor
Objective
Small       N  Median  Ave Rank      Z
1          17    1148      17.6   0.09
2          17    1123      17.4  -0.09
Overall    34              17.5

H = 0.01  DF = 1  P = 0.931
H = 0.01  DF = 1  P = 0.931  (adjusted for ties)
```

Figure3. Results of Kruskal-Wallis test for small scale problems

Table4. Computational results for small scale (1)

	Number of optimal solutions		Number of better solution than GAMS		Average error of methods compared to GAMS		Average computational time (seconds)		
Machine type	SA	GA	SA	GA	SA	GA	SA	GA	GAMS
3	16	16	4	2	0.0926	0.0385	11.2078	5.9602	842.8940
5	16	8	3	1	0.0460	0.0217	17.9294	10.6741	787.989
7	11	3	3	1	0.0255	0.0570	22.6725	13.0051	661.913
10	0	0	8	6	0.0922	0.0654	34.1321	30.3644	1233.92

Table5. Computational results for small scale (2)

Method	**Objective function average value**	**Computational time average value**
GA	1494.14	15
SA	1454.45	44.47
GAMS	1468.92	881.68

Table6. Computational results for small scale (3)

Mutual comparison of methods	Percent of problems that (1) excels (2) considering time	Percent of problems that (1) excels (2) considering objective value	Percent of problems that (1) and (2) are equal considering time
(1)SA, (2)GAMS	61.11	12.5	28.86
(1)GA, (2)GAMS	74.3	6.94	18.75
(1)SA, (2)GA	25	75.69	16.67

We can see that in the large scale problems, the proposed SA outperforms the GA in most cases. Also, it can be seen that in most cases, the time taken to reach optimal solution in proposed SA is less than the proposed GA. This leads the proposed SA to be more efficient in solving the large scale problems (except the sizes 5, 9, and 13).Considering the objective function values, we checked the normality of the mentioned data at 95% confidence level according to the results. In large scale problems, the p-value was more than 0.05 which indicates that the results are normal. For this reason (having normality data), we use one-way analysis of variance (ANOVA) to examine the equality of means obtained from the proposed algorithms. The results of this test can be seen in Table 7.

According to Figures 4 and 5, the P-VALUE = 0.558, the equality assumption of variances of the data obtained by the proposed algorithms cannot be rejected at the 95% confidence level.

Due to the Figure 6 and P = 0.628, we can conclude that equality assumption of the values resulted from the proposed algorithms (in large scale problems) cannot be rejected at a confidence level of 95%. The results of this test can be seen in Tables 8 and 9.

Table7. Computational results for large scale problems for SA and GA

Parameters				SA		GA		Gap
				Solution	Time	Solution	Solution	
N=15	*M=3*	*s1*	*f=1.5*	4080.75	18.3	4520.95	15.975	0.1078
N=20	*M=3*	*s3*	*f=1.6*	1869.5	30.68	2264.1	50.18	0.2110
N=25	*M=3*	*s1*	*f=1.3*	26995.1	32.91	27907.75	77.442	0.0338
N=30	*M=3*	*s2*	*f=1.5*	36204.05	39.64	36854.4	99.32	0.0179
N=15	*M=3*	*s3*	*f=1.6*	2321.05	39.21	2584.6	27.59	0.1135
N=20	*M=5*	*s3*	*f=1.6*	3145.15	49.96	3880.65	84.14	0.2338
N=25	*M=5*	*s1*	*f=1.3*	29801.65	51.9	31297.45	112.58	0.0501
N=30	*M=5*	*s2*	*f=1.5*	29079.2	46.45	37583.4	115.32	0.2924
N=15	*M=5*	*s3*	*f=1.3*	4642.05	39.2	6225.15	27.24	0.3410
N=20	*M=7*	*s3*	*f=1.5*	7093.5	47.25	8386.8	82,96	0.1823
N=25	*M=7*	*s1*	*f=1.3*	31464.34	57.99	37387.8	104.35	0.1882
N=30	*M=7*	*s2*	*f=1.5*	27116.65	64.24	32174.94	125.93	0.1865
N=15	*M=7*	*s3*	*f=1.6*	5075.55	58.54	5450.25	43.28	0.0738
N=20	*M=10*	*s3*	*f=1.6*	10518.15	68.44	13351.45	124.91	0.2693
N=25	*M=10*	*s1*	*f=1.5*	19535.35	76.14	22590.65	142.1	0.1563
N=30	*M=10*	*s2*	*f=1.5*	48426.91	84.94	59651.2	162.23	0.2317

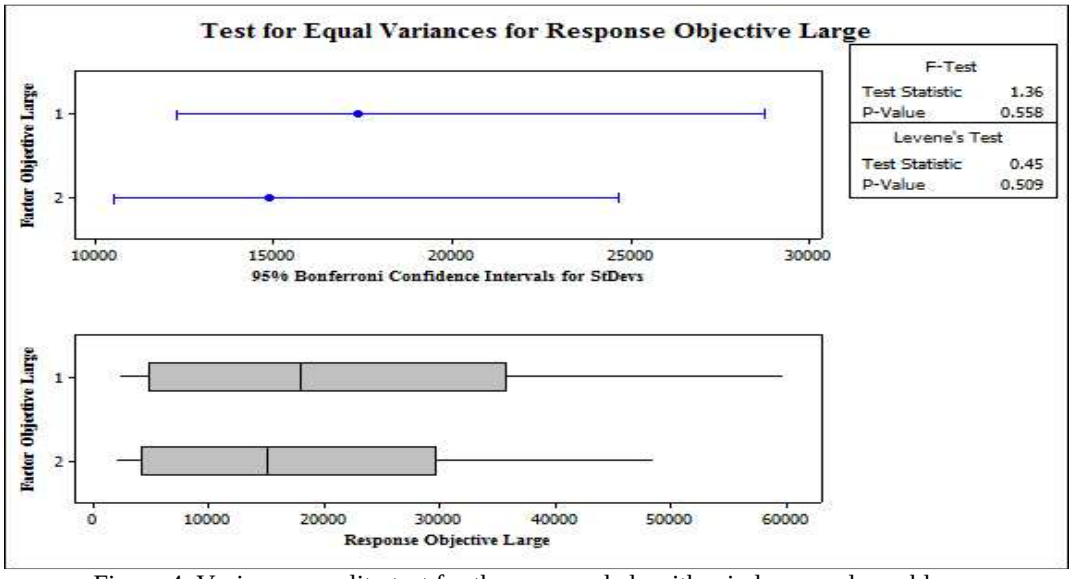


Figure 4. Variance equality test for the proposed algorithm in large scale problems

Paired T-Test and CI: GA Objective Large; SA Objective Large

```
Paired T for GA Objective Large - SA Objective Large

                     N   Mean  StDev  SE Mean
GA Objective Large  16  20757  17323     4331
SA Objective Large  16  17961  14850     3712
Difference          16   2796   3266      816


95% CI for mean difference: (1056; 4537)
T-Test of mean difference = 0 (vs not = 0): T-Value = 3.42  P-Value = 0.0
```

Figure5. T-test for the data obtained from the proposed algorithms

One-way ANOVA: Response Objective Large versus Factor Objective Large

```
Source                   DF          SS          MS     F      P
Factor Objective Large    1    62559355    62559355  0.24  0.628
Error                    30  7809240071   260308002
Total                    31  7871799426

S = 16134   R-Sq = 0.79%   R-Sq(adj) = 0.00%


                          Individual 95% CIs For Mean Based on Pooled StDev
Level   N   Mean  StDev     -+---------+---------+---------+--------
1      16  20757  17323           (----------------*----------------)
2      16  17961  14850     (----------------*----------------)
                           -+---------+---------+---------+--------
                         10000     15000     20000     25000

Pooled StDev = 16134
```

Figure6. Results of ANOVA analysis for the data obtained from large scale problems

Table8. Computational results for large scale (1)

Method	Objective function average value	Computational time average value	Percent of better solutions considering time	Percent of better solutions considering objective function value
GA	18690.68	87	3.48	25
SA	15378.22	49.96	96.52	75

Table9. Computational results for large scale (2)

Machine type	percent of better solution obtained by SA compared to GA	Average error of GA compared to SA	Average computational time (seconds)	
			GA	SA
3	86.11	0.1070	30.94	55.32
5	100	0.2611	45.50	71.88
7	100	0.2873	51.99	77.46
10	100	0.2997	71.41	117.65

7. Conclusion

In this paper, we present a simulated annealing (SA) and a genetic algorithm (GA) based on heuristics for scheduling problem of jobs in virtual cellular manufacturing systems. Computational results show that both SA ad GA algorithms perform properly but SA is likely to turn out well in finding better solutions in shorter times especially in large scale problems. For future work, these methods can be compared to some other layouts such as cellular manufacturing or process layout in

order to define the efficiency of the model. In this article, all of the variables have deterministic values. Since in the real world, the nature of variables is not deterministic, so the fuzzy approach can be applied to the problem.

8. References

[1] Greene, T. J. and Sadowski, R. P. 1984. A review of cellular manufacturing assumptions, advantages and design techniques. Journal of Operations Management, 4(2), 85-97.

[2] Wemmerlöv, U. and Hyer, N. L. 1989. Cellular manufacturing in the US industry: a survey of users. The international journal of production research, 27(9), 1511-1530.

[3] Morris, J. S. and Tersine, R. J. 1990. A simulation analysis of factors influencing the attractiveness of group technology cellular layouts.Management Science, 36(12), 1567-1578.

[4] McLean, C. R., Bloom, H. M. and Hopp, T. H. 1982. The virtual manufacturing cell. In Proceedings of Fourth IFAC/IFIP Conference on Information Control Problems in Manufacturing Technology, October, 105-111.

[5] Slomp, J., Chowdary, B. V. and Suresh, N. C. 2005. Design of virtual manufacturing cells: a mathematical programming approach. Robotics and Computer-Integrated Manufacturing, 21(3), 273-288.

[6] Kesen, S. E., Toksari, M. D., Güngör, Z. and Güner, E. 2009. Analyzing the behaviors of virtual cells (VCs) and traditional manufacturing systems: ant colony optimization (ACO)-based metamodels. Computers and Operations Research, 36(7), 2275-2285.

[7] Kesen, S. E., Das, S. K. and Güngör, Z. 2010. A genetic algorithm based heuristic for scheduling of virtual manufacturing cells (VMCs). Computers and Operations Research, 37(6), 1148-1156.

[8] Kesen, S. E., Das, S. K. and Gungor, Z. 2010. A mixed integer programming formulation for scheduling of virtual manufacturing cells (VMCs). The International Journal of Advanced Manufacturing Technology, 47(5-8), 665-678.

[9] Mak, K. L., Peng, P., Wang, X. X. and Lau, T. L. 2007. An ant colony optimization algorithm for scheduling virtual cellular manufacturing systems. International Journal of Computer Integrated Manufacturing, 20(6), 524-537.

[10] Mak, K. L., Ma, J. and Su, W. 2010, A constraint programming approach for production scheduling of multi-period virtual cellular manufacturing systems. In Natural Computation (ICNC), Sixth International Conference on, August, 8, 4440-4444.

[11] Babu, A. S., Nandurkar, K. N. and Thomas, A. 2000. Development of virtual cellular manufacturing systems for SMEs. Logistics Information Management, 13(4), 228-242.

[12] Khilwani, N., Ulutas, B. H., Islier, A. A. and Tiwari, M. K. 2011. A methodology to design virtual cellular manufacturing systems. Journal of intelligent manufacturing, 22(4), 533-544.

[13] Mahdavi, I., Aalaei, A., Paydar, M. M. and Solimanpur, M. 2011. Multi-objective cell formation and production planning in dynamic virtual cellular manufacturing systems. International Journal of Production Research, 49(21), 6517-6537.

[14] Kannan, V. R. 1997. A simulation analysis of the impact of family configuration on virtual cellular manufacturing. Production Planning and Control, 8(1), 14-24.

[15] Nikoofarid, E. and Aalaei, A. 2012. Production planning and worker assignment in a dynamic virtual cellular manufacturing system. International Journal of Management Science and Engineering Management, 7(2), 89-95.

[16] Hamedi, M., Esmaeilian, G. R., Ismail, N. and Ariffin, M. K. A. 2012. Capability-based virtual cellular manufacturing systems formation in dual-resource constrained settings using Tabu Search. Computers and Industrial Engineering, 62(4), 95.