J. King Saud Univ., Vol. 20, *Eng. Sci.* (2), pp. 83-94, Riyadh (2008/1429H.)

An Optimization Model and GA for Solving Machine and Process Selection Problems in Manufacturing Systems

Abdulrahman M. A. Al-Ahmari and Mohamed Z. Ramadan
Industrial Engineering Department, College of Engineering, King Saud University,
P.O. Box 800, Riyadh 11421, Saudi Arabia
Email: alahmari@ksu.edu.sa

(Received 20 September 2006; accepted for publication 27 February 2007)

Keywords: Machine selection, Process selection, Dynamic manufacturing systems, Genetic algorithms.

Abstract. In some industries such as electronic industries, a dynamic manufacturing environment is adopted. In these systems, manufacturing processes and machines selection is based upon the released orders over specific time periods. Therefore, each order is characterized by a part mix that may need to be revised based on machine availability or other economical factors. However, it is necessary to consider the costs of acquiring new machines and revising manufacturing process and to evaluate the benefits derived after such changes of this dynamic system. Chen (1999) considered this problem and suggested a heuristic algorithm based on a decomposition method as a solution procedure. This paper presents the optimal solutions to the machine and process selection problem and suggests a Genetic Algorithm (GA) to solve such problems. The obtained results are evaluated and compared against the existing methods using numerical examples.

1. Introduction

The main characteristic features of today's dynamic manufacturing environments are apparent and they can be listed as follows: stochastic demand, variable but smaller production batch size, frequent and unpredictable changes in product mix, highly variable processing and set-up times, variable production sequences, very high volume of information and a strong competition (Saad, 2003). Researchers have considered a number of technological and operational problems for designing and setting up the dynamic manufacturing systems. These problems include machine selection, part selection, resource allocation, loading problems, planning and scheduling. In this paper, a dynamic manufacturing system is considered. In such systems, more frequent system changes are possible based upon several factors including machine availability, required part mix, capacities, etc.

Esawi and Ashby (1998) discussed two steps that involved in manufacturing process selection. The first step involves the screening of all available processes to determine whether they are technically capable of manufacturing the design; the second involves the ranking of those which are successful using economic criteria. The ranking step requires the techniques of cost estimation. In seeking to achieve this, two problems are encountered. First, design is still in an early stage at which little information is available; and second, conventional cost estimation techniques require detailed information and cannot easily be applied to widely diverse processes. Yu *et al.* (1992) proposed a methodology for process selection that can be applied to early stages of product design. They focused on net-shape manufacturing processes and identified the major factors that affect the selection of an appropriate process. Chen (1999) developed an integer programming model considering the equipment selection problem with multiple manufacturing process alternatives and multiple time periods. Operations of the parts considered in this model can be processed by more than one machine due to machine flexibility. The objective of the model is to minimize the overall manufacturing process costs for acquiring and installing machine tools with

various production demands in several periods. He solved this model using a heuristic algorithm. Chen *et al.* (1995) developed a 0-1 integer programming model to formulate the part selection problem in FMS. They developed two heuristic algorithms and three prioritizing strategies to solve the part selection problem efficiently as well as effectively. Performances of all the heuristic solution methods were evaluated based on the profits generated from producing the set of parts selected.

Lee *et al.* (1997) developed three iterative algorithms, called the forward algorithm (FWA), the backward algorithm (BWA) and capacity approximation algorithm (CAA) that solve the part selection and loading problems iteratively for each period. FWA solves the part selection and loading problems iteratively from the first period of the planning horizon to the last, one by one, while BWA solves them in the reverse direction. On the other hand, CAA first selects parts for each period up to the capacities considering subcontracting the costs of the parts. Liang and Taboun (1992) developed a model which can concurrently handle part selection and part (operation) assignment problems. The uniqueness of their model lies in that it captures the cellular layout feature of many existing FMSs when dealing with part selection problem and simultaneously solves part selection and part assignment problems. The model was solved using LINDO package. Kusiak and Finke (1988) considered the selection of process plans in automated manufacturing systems by formulating a model with the objective of minimizing cost and usage of the number of tools and auxiliary devices. Al-Ahmari (2002) developed two mathematical models for the selection of machines and part types in cellular manufacturing systems. The first model identifies machine groups, cutting speeds, number of machine types required and a process plan for each part assuming the part families are known. The second model is used to determine the part families, machine groups, number of machines, plan for each part, and the cutting speed for each operation selected. The second model identifies these variables simultaneously. Tiwari and Vidyarthi (1998) considered the plan selection problem in an automated manufacturing system, taking into account the similarity measures among the process plans of the parts. Four algorithms have been developed to integrate several segments of the process plan selection problem. Lee and Kim (2000) considered a loading problem for the partial grouping configuration in flexible manufacturing systems with the objective of minimizing the maximum workload of the machines. Several heuristic algorithms were suggested for the loading problem. They compared the suggested algorithms against an existing one using computational experiments which were done on randomly generated test problems. In addition, they used simulation experiments to investigate the effects of grouping methods and loading algorithms on system performance. Jang *et al.* (1996) developed an integrated decision for FMS planning and scheduling problems (FMSDS) that can solve the sub-problem such as the routing problems, the part type selection problem, the tool allocation and order adjusting problems, and scheduling problems. Almutawa *et al.* (2005) presented an approach for optimizing the number of machines acquired for batch processing in a multistage manufacturing system with different processing capabilities. They examined four cases of overlap in machine capabilities to find the optimum number of machines of each type to be purchased for each stage, as well as the optimum time delay between the stages when the system cost is minimized. They simulated the resulting multistage line under specified conditions to observe its behavior over time and to assess the viability of the developed model. Liang and Dutta (1992) investigated the combined part selection (PS), loading sharing (LS), and machine loading (ML) problems in hybrid manufacturing system (HMS). Their objective is to obtain consistent solutions to the related decisions by concurrently considering the three problems and to improve the overall system performance by coordinately utilizing the production resources of flexible manufacturing system (FMS) and conventional manufacturing system (CMS). In another paper, Liang and Dutta (1993) proposed a sequential-bicriteria approach to solving concurrently part selection, machine-loading and tool-configuration problems in a class of flexible manufacturing systems. Associated models were developed with example to illustrate their application. The model can be applied only when enough copies of each type of tool are available. They have also suggested a solution method for larger problems. Many others have considered process and part selections in static manufacturing systems. Recently, Sujono and Lashkari (2007) proposed a method for simultaneously determining the operation allocation and material handling system selection in a flexible manufacturing system environment with multiple

performance objectives. They developed an integer programming model to select machines, assign operations of part types to the selected machines, and assign material handling equipment to transport the parts from machine to machine, as well as to handle the part at a given machine. The selection is based on the compatibility between the material handling equipment and the parts. Very few papers considered such problems in the dynamic manufacturing systems such as Chen (1999).

In this paper, the optimal solution of machine and process selection problem model is obtained. A GA is also developed to solve the presented problem. The obtained optimal results are used to compare and evaluate the results of the suggested GA and Chen's (1999) results. The rest of this paper is organized as follows: Section 2 presents the problem description and model development; Section 3 provides the numerical example and the optimal solution; Section 4 describes the details of the suggested GA; Section 5 presents more computational results; and finally, Section 6 concludes the paper and gives guidelines for future research.

2. Problem Description and Model Development

In a dynamic manufacturing system, part mix and system configuration is changed over the different planning periods. Chen (1999) described the problem of machine and process selection in manufacturing systems over a number of time periods. The system consists of a number of different machines which are used to process a number of different parts in batches with a certain number of operations. Each operation may be processed by one machine or other available machines in the considered system taking into account the cost and capacity required for an operation corresponding to the machine used to perform the operation. Chen (1999) suggested an integer programming model to minimize the processing cost and the costs for installing and operating the machines for the entire planning time horizon, subject to part production processes and requirements. The notation and the model suggested by Chen (1999) are given below:

Ak cost to remove one type *k* machine from the system.

Ak cost to install one type *k* machine in the system.

B_k maximum number of type *k* machines for time period *t*.

$c_{ijk}(t)$ cost to process operation *j* of part *i* using machine *k* at time *t*.

Decision variables to be used in the model are:

$f_i(t)$ units of part *i* to be produced in time *t*.

H_k cost of having one type *k* machine in the system for time period *t*.

i part index, *i* 1,. . . , *I*.

j index of operations of part *i*, *j* 1,. . . , *Ji*.

k machine index, *k* 1,. . . , *K*.

$M_k(t)$ number of type *k* machines to be installed in the system during time period *t*.

P_s population size.

$Q_{ijk}(t)$ capacity requirement for type *k* machine (in terms of one unit of type *k* machine) by operation *j* of part *i* at time *t*.

t time index, *t* 1,. . . , *T*.

$x_{ijk}(t)$ 1, if operation *j* of part *i* will be processed by machine *k* during time *t*; 0, otherwise,

yk t number of type *k* machines removed from the system at the end of time *t*.

yk t number of type *k* machines added to the system at the end of time *t*.

The objective function of the model is expressed as:

Min $Z(\mathbf{x}(t),\mathbf{M}(t))=$

$$\sum_{t=1}^{T}\sum_{i=1}^{I} f_i(t)\sum_{j=1}^{J}\sum_{k=1}^{K} C_{ijk}(t) * X_{ijk}(t) + \sum_{t=1}^{T}\sum_{k=1}^{K} H_k(t) * M_k(t) + \sum_{t=1}^{T-1}\sum_{k=1}^{K}\left[A_k^{+}(t) * Y_k^{+}(t) + A_k^{-}(t) * Y_k^{-}(t)\right] \quad (1)$$

Subject to:

$$\sum_{k=1}^{K} X_{ijk}(t) = 1,\ j = 1,....,\ J_i,\ i = 1,.....,I,\ t = 1,.....,T \quad (2)$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J_i} Q_{ijk}(t) * X_{ijk}(t) \le M_k(t),\ k = 1,....,K,\ t = 1,.....,T \quad (3)$$

$$M(t) \le B(t),\ k = 1,....,K,\quad t = 1,.....,T \quad (4)$$

$$M(t+1) = M(t) + [Y(t) - Y(t)],\ k = 1,....,K,\ t = 1,.....,T \quad (5)$$

$$X_{ijk}(t)=0,1,\forall i,j,k,t \quad (6)$$

$$M_k(t),Y_k^{+}(t),Y_k^{-(t)}\geq 0 \quad and \quad Integer,$$
$$k=1,.....,K,\ t=1,.....,T \quad (7)$$

As shown in Eq. (1), the objective function consists of three terms. The first term is used to calculate the processing costs, the second term is to calculate the cost of having the machine in the system for a specific time period, and the third term is to calculate the installation/removal costs of machines into/from the system. Constraint (2) is to ensure that operation *j* of part *i* will be processed by only one type of machine in the system. Constraints (3) and (4) are to make sure that there is enough machine capacity to process the parts. Constraint (5) defines the number of machines installed in the next period. Constraints (6) and (7) are for binary and non-negativity.

Chen (1999) solved the above model using a decomposition-based heuristic method, but he did not get the optimal solution. In this paper, we coded the model using LINDO software and the optimal solution is obtained. Solving such a model became possible with the advent of high-speed computers and highly sophisticated optimization software. Also, we developed a GA to solve the process and machine selections, as discussed later in this paper.

3. Numerical Example

This example is taken form Chen (1999). In this example, there are three different types of machines to process 12 types of parts in 3 time periods (*t= 1, 2, 3*). Four part types are processed in each time period with 2 to 4 operations. An operation may be processed by machine 1, machine 2 or machine 3 taking into account machine capacities and operation costs. These example data for the 3 time periods are shown in Tables 1, 2 and 3. As shown in Table 1, in period 1, part 1 has 2 operations and operation 1 requires 1.0 unit of machine 1 with processing cost being $8.0. This operation of part 1 can also be processed by machine 2, requiring 1.3 units to complete with a cost of $9.0. Table 4 illustrates the maximum number of machines, unit machine holding costs, and machine installation/removal costs for the three types of machines over the three time periods.

Based on the given data, the LINDO model is developed and solved. The optimal solutions are obtained for the three time periods, as illustrated in Tables 5-7. The optimal results of machines and costs are shown in Table 8. Since the problem is NP-hard and has large number of integer variables, we tried to reduce the computation time by developing an efficient GA procedure.

4. Genetic Algorithm

Genetic Algorithm (GA) theory and procedures are addressed in several books (e.g., (Goldberg, 1989; Holland, 1975)). Michalewicz (1992) stated that genetic algorithm for a particular problem must have the following five components:

1. A genetic representation of potential solutions to the problem.
2. A way to create an initial population of potential solutions.
3. An "Evaluation Function" rating in terms of its "Fitness".
4. Genetic operators.
5. Values of various "Parameters" (population size, and the probabilities of applying each operator).

In the selection problem of machines and operations, the representation of the model parameter set as a binary form (binary string {0,1}) of arbitrary length as illustrated in Fig. 1.

In our example, the chromosome is of a length equal to 16 bits. A gene in this case can be 0 or 1 and the binary string of length n can represent the multiplication of the number of intervals, number of produced parts, number of required operations, and the number of available machines, as shown in Fig. 1. The size of string length, in this figure, is 16 bits, which is based on the double of each element. A bit "1" means the machine is assigned to perform an operation on part type at that period of time. A bit "0" represents that machine is not selected.

A population consists of a number of individuals being tested (search space). The population is initialized with randomly valued individuals. The size of population may be fixed or changeable and is set by the user to suit the requirements of the particular model. In the considered example, the size of the population is set changeable based on 10% of the multiplication of number of intervals, number of produced parts, number of required operations, and the number of available machines. The initialization process involves the creation of a population of chromosome. All the bits for each chromosome are

Table 1. Part production data of the example (t=1)

Part	**1**				**2**					**3**							**4**										
Operation	1		2		1		2		3	1			2			3			4		1		2			3	
Machine	1	2	2	3	1	3	1	2	2	3	1	2	3	2	3	1	2	3	2	3	1	3	2	3	1	2	3
Processing Cost	8.0	9.0	7.0	6.0	5.0	3.0	4.0	9.0	2.0	1.0	8.5	9.0	7.2	4.3	3.1	6.5	4.0	2.2	2.8	3.5	9.7	4.8	5.6	3.9	2.3	6.7	5.2
Required Capacity	1.0	1.3	0.4	0.7	0.5	0.7	0.7	0.5	0.6	0.4	1.2	1.6	1.4	0.7	1.1	1.4	1.0	1.2	0.8	0.5	1.5	1.8	1.4	1.1	1.3	1.2	1.0

Table 2. Part production data of the example (t=2)

Part	**1**					**2**					**3**							**4**									
Operation	1		2		3		1		2		1		2		1		2		3		4						
Machine	1	2	2	3	1	3	2	1	2	2	3	1	2	3	2	3	1	2	2	3	1	3	2	2	3		
Processing Cost	7.0	6.0	5.0	8.0	4.0	5.0	6.0	6.0	5.5	2.0	3.0	4.5	4.0	5.2	3.3	4.1	3.5	4.0	4.8	3.5	5.7	6.8	4.2	3.5	2.7		
Required Capacity	1.0	1.3	0.4	0.7	0.5	0.7	0.3	0.7	0.5	0.6	0.4	1.2	1.6	1.4	0.7	1.1	1.4	1.0	0.8	0.5	1.5	1.8	1.6	1.3	1.0		

Table 3. Part production data of the example (t=3)

Part	**1**				**2**					**3**							**4**									
Operation	1		2		1		2		3	1			2		1		2		3								
Machine	1	3	2	3	1	1	2	1	2	3	2	1	1	2	3	1	3	1	2	2	3	1	3	2			
Processing Cost	8.0	9.0	7.0	6.0	7.5	5.0	3.0	4.0	9.0	6.6	2.0	1.0	8.5	9.0	7.2	4.3	3.1	6.5	4.0	2.8	3.5	9.7	4.8	6.0			
Required Capacity	1.5	2.1	0.7	0.9	0.5	0.8	1.7	0.9	0.7	0.8	0.6	0.8	1.4	1.8	1.6	0.9	1.4	1.6	1.2	1.0	0.8	1.7	2.0	1.4			

Table 4. Machine limits, holding and other costs of the example

Machine Type	t=1				t=2				t=3	
	M.L.	H.C.	I.C.	R.C.	M.L.	H.C.	I.C.	R.C.	M.L.	H.C.
1	4	12	3	2	6	14	4	3	3	16
2	5	15	3	2	6	18	4	3	6	20
3	6	10	3	2	6	12	4	3	4	15

M.L.: Machine limit.
H.C.: Machine holding cost.
I.C.: Machine holding cost.
R.C.: Machine removing cost.

Table 5. Optimal solution for *t*=1

Part	1		2			3				4		
Operation	1	2	1	2	3	1	2	3	4	1	2	3
Machine	1	2	3	1	3	1	2	2	2	3	3	3
Processing Cost	8.0	7.0	3.0	4.0	1.0	8.5	4.3	4.0	2.8	4.8	3.9	5.2
Required Capacity	1.0	0.4	0.7	0.7	0.4	1.2	0.7	1.0	0.8	1.8	1.1	1.0

Table 6. Optimal solution for *t*=2

Part	1			2			3			4	
Operation	1	2	3	1	2	1	2	1	2	3	4
Machine	2	2	3	2	3	3	2	1	3	1	3
Processing Cost	6.0	5.0	5.0	5.5	3.0	5.2	3.3	3.5	3.5	5.7	2.7
Required Capacity	1.3	0.4	0.7	0.5	0.4	1.4	0.7	1.4	0.5	1.5	1.0

Table 7. Optimal solution for *t*=3

Part	1		2			3			4	
Operation	1	2	1	2	3	1	2	1	2	3
Machine	1	1	1	2	2	3	3	2	3	2
Processing Cost	8.0	7.5	5.0	9.0	2.0	7.2	3.1	4.0	3.5	6.0
Required Capacity	1.5	0.5	0.8	0.7	0.6	1.6	1.4	1.2	0.8	1.4

Table 8. The optimal results on machines and costs

Time	*t = 1*	*t = 2*	*t = 3*	*Total*
Machine 1	3	3	3	8
Machine 2	3	3	4	9
Machine 3	5	4	4	12
Machine Cost	131	144	188	463
Processing Cost	56.5	48.4	55.3	160.2
System Change Cost	2	4	n/a	6
Total Cost	189.5	196.4	243.3	629.2

Period 1								Period 2……
Part 1				Part 2				
Operation 1		Operation 2		Operation 1		Operation 2		……..
M/C1	M/C2	M/C1	M/C2	M/C1	M/C2	M/C1	M/C2	…….....
1	0	0	1	1	0	0	0	…….....

Fig. 1. One of the chromosome representations.

initialized randomly. In addition, all chromosomes are tested against the maximum availability of each type of machines.

The evaluation function is the objective function that measures the fitness of the chromosome, which is the value of the objective function for its phenotype. Therefore, to calculate the fitness, the chromosome must firstly decode into its phenotype to be tested against the evaluation function. For each individual chromosome Xi (i = 1, 2, . . . , Ps), the locally minimized total costs (e.g., including processing, installation, removing, and holding cost) is obtained.

Among the population, the candidate is characterized by the fitness, which is used as the probability of choosing that individual as a parent. The next generation, children are obtained from the parents of the current generation by choosing one of the genetic operators. The selection of a new population with respect to the probability distribution is based on fitness value. A roulette wheel selection method is used as follows:

a. Calculate the fitness value, FIT(ci), for each chromosome Ci (i = 1, 2, …..,Ps).

$$C_T = \sum_{i=1}^{P_s} FIT(c_i) \tag{8}$$

b. Find the total fitness of the population, CT.
c. Calculate the probability of selection, Pi, for each chromosome Ci (i = 1, 2, …, Ps):

$$P_i = \frac{FIT(c_i)}{C_T} \tag{9}$$

d. Calculate the cumulative probability, CPi, for each chromosome Ci (i = 1, 2,…….., Ps):

$$CP_i = \sum_{j=1}^{i} P_j \tag{10}$$

e. Spinning the roulette wheel pop-size times. Each time a single chromosome is selected for a new population as follows:
f. Generate a random float number R from the range (0, 1).
g. If R < CP1 then select the first chromosome, C1; otherwise select the i^{th} chromosome, C_i (where $2 \le i \le Ps$), such that $CPi -1 \le R \le CPi$.

Genetic operators are the hearts of genetic algorithm in that they are considered the exploration tools that search the alternative space for hunting better quality solutions. By means of these genetic operators, some members (chromosomes) of the population undergo transformation (alteration) to form new solutions (offspring). After some number of generations, it is hoped that the best individual represents a near optimum reasonable solution. Three genetic operators, that are most common, are implemented. These operators are "crossover", "mutation" and "reproduction".

Crossover operator is implemented in two steps. First, the members of the newly reproduced strings in the population are mated at random. An expected number, equal to the probability of crossover, PC, multiplied by pop-size, are selected to undergo crossover operation:

a) Generate a random float number R from the range (0, 1); and
b) If R < PC, select a given chromosome for crossover; otherwise a next chromosome is considered and return to step a.

Second, each pair of strings (parents) undergoes crossing over as follows: two integer positions, between one and the string length multiply by the number of machines, and along the string, are selected uniformly at random. These positions divide the parents into three segments. Two new strings (children or offspring) are created by exchanging some group of characters within the produced segments in some way depending upon the crossover technique. For instance, suppose that we have mated parents in a binary form, string 1 and string 2. For double crossing position (two site splices), the crossover operator may be accomplished as illustrated in Fig. 2.

The function of the mutation operator is to introduce some variability, at random, that disturbing

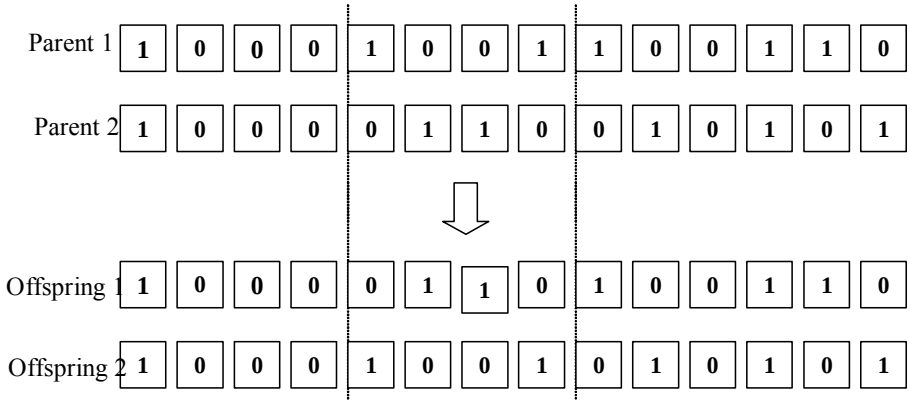

Fig. 2. Double position crossover.

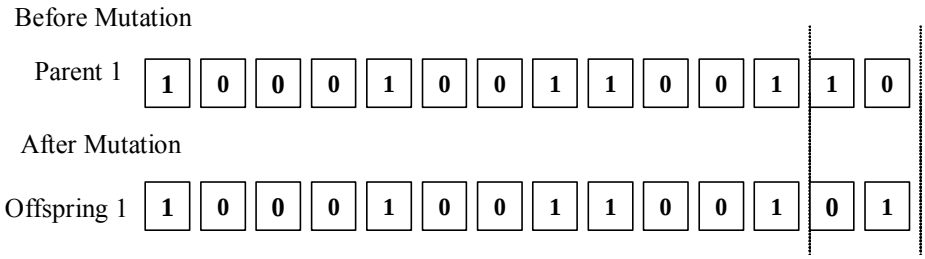

Fig. 3. Mutation operator.

genetic information to emulate what is happening in the natural genetic process. Mutation is performed on a bit-by-bit basis, in that it alters one or more genes (positions) in chromosome, with a probability equal to the mutation rate, PM. This probability (PM), which is one of the genetic algorithm parameters, is set by the user and usually to a quite small value about "0.01". The mutation probability gives us the expected number of bits (genes) undergoing mutation per generation, which is equal to PM multiplied by the population size (Ps) and the string length (n). In this problem, for example, we have a binary chromosome that has a last bit (e.g., a machine) selected randomly to undergo mutation, as shown in Fig. 3. An expected number, equal to the probability of mutation P multiplied by string length by the population size, are selected to undergo mutation operation. The mutation performed on a bit of machines basis:

a) Generate a random float number R from the range (0, 1).
b) If R < PM, mutate the bit; otherwise select the next bit and return to step a.

Reproduction is a process in which individual strings are copied according to a selection method based on their objective function values (fitness function). Copying strings according to their fitness values mean that the string with higher value has a higher probability of contributing one or more offspring in the next generation. This operator is of course an artificial version of natural selection, a survival of the fittest among string creatures (organisms). Here, the objective function is the final step of the string creature's life or death.

The reproduction operator may be implemented in an algorithmic form and in number of ways. One of the traditional genetic algorithm techniques is the roulette wheel selection method, which is the easiest way to create a biased operation toward the fittest chromosomes. Every current string in the population has a roulette wheel slot size in proportion to its fitness. In this process, the fitness over all strings is summed to obtain the grand total value. Then, the percentage of the population total fitness is calculated for each string to represent the probability of each one to be selected to the next generation. The roulette wheel is spun population size times. Thus, the selection of the new population is dependant upon the probability distribution based on fitness value in that each time spinning roulette wheel, a single

chromosome is selected for a next generation providing its probability exceeds a generated random number. Obviously, in this way, more highly fit strings will have a higher number of offspring in the succeeding generation and therefore the best chromosome gets more copies, the average stay even, and the worst die off (Michalewicz, 1992).

The parameters of GAs are: population size, and probability of applying each operator (e.g., crossover, mutation and reproduction). As mentioned previously, the population size is set by the user to suit the requirements of a particular model. Running the algorithms with large population size can be expensive in terms of time, so a smaller population may be desirable, but if the population is too small, then the loss of genetic diversity may compromise the search. Genetic diversity in GAs is important when the solution space is rugged or convoluted. A small population would be more likely to quickly converge on what may be local optima. Also, with the sparse spread of initial points, better optima may never be visited before the search converges on a poor local optima, or even if they are visited, the point may be comparatively unfit, and will not be given an adequate opportunity to reproduce and start hill climbing. At the other extreme, there are problems concomitant with excessively large population. Firstly, the initialization becomes expensive. In some cases, the initialization of a large population is equivalent to a random search and produce optimal or near optimal solution, which makes the use of GAs somewhat moot. After all of these, population size must be selected to be adequate to the specific problem at hand as well as it must be based on experimentation. Therefore, the population size was set as 10% of the multiplication of the number of intervals, number of produced parts, number of required operations, and the number of available machines. Also, the probability of applying each operator is an important parameter to the algorithm. The probability of crossover gives us the expected number of individuals that undergo crossing over per generation. The cross over operator is an important search tool that explores the search space, and exchange notion between individuals in an attempt to hunt the good quality characteristic. A high probability of crossover maybe beneficial, but it can also be expensive in terms of time and can ensue in missed-good characteristics. Other important parameters of genetic algorithm are the probabilities of mutation and reproduction. These probabilities are suggested to be small, as their function is to introduce some variability after the crossover operator in order to imitate what is happening in the natural genetic process. However, all of these parameter values are set by the user according to the particular problem at hand and that these values should be subjected to experiments to identify its adequate (optimal) values. In this work, the probabilities of crossover, mutation and reproduction were set as 65%, 10% and 5%, respectively.

Following the selection, crossover, mutation and the processes, the new population is ready for evaluation. This evaluation is used to build the probability distribution (i.e., construction of roulette wheel) for the next selection process. The whole process is repeated for a fixed number of iterations (generations) depending on the speed and resource criteria. In this work, for example, 100 of the iterations were selected. In each generation, the best-fit solution is selected and stored.

The results obtained by the developed GA are shown in Tables 9-11 for the three time periods. Table 12 illustrates the number of machines required in the three time periods and the different cost figures associated with the overall solution.

The results obtained from both optimal model and GA, are compared against Chen's results, as illustrated in Table 13. In this example, GA results are closer to the optimal results than Chen's results.

5. More Computational Results

Other computations are conducted for large sized problems and solved using the optimal model and the suggested GA. Table 14 summarizes the obtained results for these examples. This table illustrates the number of time periods, machines, part types and operations. The relative errors are calculated to evaluate the accuracy of the developed GAs for these considered examples, as follows:

$$\text{Relative error} = \frac{\text{(Objective function value of the optimal solution)-(Objective function value of the GA)}}{\text{(Objective function value of the optimal solution)}}$$

It is clear from the above table that the suggested GA can generate very good quality solutions for various size of selection of machines and operations problems in manufacturing systems. The relative errors of the

Table 9. GA solution for *t*=1

Part	1			2			3				4	
Operation	1	2	1	2	3	1	2	3	4	1	2	3
Machine	1	2	1	2	3	1	2	2	3	3	3	3
Processing Cost	8.0	7.0	5.0	9.0	1.0	8.5	4.3	4.0	3.5	4.8	3.9	5.2
Required Capacity	1.0	0.4	0.5	0.5	0.4	1.2	0.7	1.0	0.5	1.8	1.1	1.0

Table 10. GA solution for *t*=2

Part	**1**			**2**		**3**			**4**		
Operation	1	2	3	1	2	1	2	1	2	3	4
Machine	1	2	2	2	3	1	2	2	3	1	3
Processing Cost	7.0	5.0	6.0	5.5	3.0	4.5	3.3	4.0	3.5	5.7	2.7
Required Capacity	1.0	0.4	0.3	0.5	0.4	1.2	0.7	1.0	0.5	1.5	1.0

Table 11. GA solution for *t*=3

Part	**1**			**2**			**3**		**4**	
Operation	1	2	1	2	3	1	2	1	2	3
Machine	1	1	1	2	2	3	3	2	3	2
Processing Cost	8.0	7.5	5.0	9.0	2.0	7.2	3.1	4.0	3.5	6.0
Required Capacity	1.5	0.5	0.8	0.7	0.6	1.6	1.4	1.2	0.8	1.4

Table 12. The GA results on machines and costs

Time	***t = 1***	***t = 2***	***t = 3***	***Total***
Machine 1	3	3	5	11
Machine 2	4	3	2	9
Machine 3	3	4	4	11
Machine Cost	131	134	188	453
Processing Cost	55.29	39.46	59.11	153.86
System Change Cost	9	15	0	24
Total Cost	195.29	188.46	247.11	630.86

Table 13. Comparisons the suggested models against Chen's results

Time	***t = 1***			***t = 2***			***t = 3***			***Total***		
Solution Methods	**Optimal Results**	**Chen (1999)**	**GA Results**	**Optimal Results**	**Chen (1999)**	**GA Results**	**Optimal Results**	**Chen (1999)**	**GA Results**	**Optimal Results**	**Chen (1999)**	**GA Results**
Machine 1	3	3	3	3	4	3	3	4	5	8	10	11
Machine 2	3	3	4	3	3	3	4	4	2	9	9	9
Machine 3	5	5	3	4	2	4	4	3	4	12	9	11
Machine Cost	131	131	131	144	134	134	188	189	188	463	454	453
Processing Cost	56.5	64.2	55.29	48.4	50.2	39.46	55.3	58.8	59.11	160.2	173.2	153.86
System Change Cost	2	9	9	4	8	15	n/a	n/a	n/a	6	17	24
Total Cost	189.5	204.2	195.29	196.4	192.2	188.46	243.3	247.8	247.11	629.2	644.2	630.86

Table 14. More computational results for example problems

Example	No. of Periods	No. of Machines	Number of Parts	Total No. of Operations	Optimal Solution	GA Solution	Relative Error %
1	3	3	12	33	629.2	630.86	0.26
2	3	4	12	48	937.9	953.0	1.6
3	4	5	12	57	981.7	1021.3	4.0
4	4	5	16	77	1311.4	1368.2	4.33
5	5	5	16	80	1532.0	1609.6	5.03
6	5	6	20	93	1940.8	2033.3	4.77
7	6	6	20	101	2207.6	2322.8	5.21

considered examples are between 0.26% and 5.21%. The GA models are coded in QBASIC and run on IBM-PC (Pentium 4). For all the examples considered, the GA is able to achieve near-optimal solutions. This indicates the success of the GA model that is developed in this paper.

6. Conclusions

More frequent system changes are possible in a dynamic manufacturing system environment which requires considering the problem of selecting the best machines operations within a multiple period time horizon. This paper presents the optimal solution to the selection problem of machines and operations in a dynamic manufacturing system. In addition, GA is developed to solve such types of problems. The developed models are tested and validated using different numerical examples. It has been found that the solutions provided by the suggested GA are near optimal solutions. There are other factors which are required to be included when considering this type of dynamic manufacturing systems such as workers assignment, tool selection, part type quantities, and other factors.

Acknowledgement. The authors would like to thank King Saud University for their support. They also would like to thank the anonymous reviewers for their constructive comments on an earlier version of this paper

References

Al-Ahmari, A.M.A. "Integrated Models for Cellular Manufacturing with Alternate Process Plans and Machining Economics." *Journal of King Saud University* (*Engineering Sciences)*, Vol. 15, No. 1, (2002), 95-111.

Almutawa, S.; Savsar, M. and Al-Rashdan, K. "Optimum Machine Selection in Multistage Manufacturing Systems." *International Journal of Production Research*, Vol. 43, (2005), 1109-1126.

Chen, F.; Ker, J. and Kelawpatinon, K. "An Effective Part Selection Model for Production Planning of Flexible Manufacturing Systems." *International Journal of Production Research*, Vol. 33, (1995), 2671-2683.

Chen, M. "A Heuristic for Solving Manufacturing Process Selection and Equipment Selection Problem." *International Journal of Production Research,* Vol. 37, (1999), 359-374.

Esawi, A. and Ashby, M. "Cost-based Ranking for Manufacturing Process Selection." *Proceedings of the Second Conference on IDMME*, Compiegne, France, Vol. 4, (May 27-29, 1998), 1001-1008.

Goldberg, D.E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, Massachusetts: Addison-Wesley, (1989).

Holland, J. H. *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press, (1975).

Jang, S. Y.; Park, J. and Park, N. "An Integrated Decision Support System for FMS Production Planning and Scheduling Problems." *International Journal of Advanced Manufacturing Technology*, Vol. 11, (1996), 101-110.

Kusiak, A. and Finke, G. "Selection of Process in Automated Manufacturing Systems." *IEEE Computer Society Press*, Vol. 11, (1988), 359-364.

Lee, D. and Kim, Y. "Loading Algorithms for Flexible Manufacturing Systems with Partially Grouped Machines." *IIE Transactions*, Vol. 32, (2000), 33-47.

Lee, D.; Lim, S.; Lee, G.; Jun, H. and Kim, Y. "Multi-period Part Selection and Loading in Flexible Manufacturing Systems." *Computers Industry,* Vol. 33, (1997), 541-544.

Liang, M. and Dutta, S. "Combined Part Selection, Load Sharing and Machine Loading Problem in Hybrid Manufacturing Systems." *International Journal of Production Research,* Vol. 30, (1992), 2335-2349.

Liang, M. and Dutta, S. "Solving a Combined Part Selection, Machine Loading, and Tool-configuration Problem in Flexible Manufacturing Systems." *Production and Operations Management*, Vol. 2, (1993), 97-113.

Liang, M. and Taboun, S. "Part Selection and Part Assignment in Flexible Manufacturing Systems with Cellular Layout." *Computers and Industrial Engineering*, Vol. 23, (1992), 63-67.

Michalewicz, Z. *Genetic Algorithms + Data Structure = Evolution Programs*. New York: Springer-Verlag, (1992).

Tiwari, M. and Vidyarthi, K. "An Integrated Approach to Solving the Process Plan Selection Problem in an Automated Manufacturing System." *International Journal of Production Research,* Vol. 36, (1998), 2167-2184.

Yu, J.; Krizan, S. and Ishii, K. "Computer Aided Design for Manufacturing Process Selection." *Journal of Intelligent Manufacturing*, Vol. 4, (1993), 199-208.

أنموذج أمثلية وأنموذج جيني لحل مشكلة اختيار الآلة والعملية في نظم التصنيع

عبدالرحمن الأحمري ومحمد رمضان

قسم الهندسة الصناعية، كلية الهندسة،

جامعة الملك سعود، ص ب ٨٠٠، الرياض ١١٤٢١،

المملكة العربية السعودية

(قدم للنشر في ٢٠٠٦/٠٩/٢٠م؛ وقبل للنشر في ٢٠٠٧/٠٢/٢٧م)

ملخص البحث. يطبق نظام التصنيع الديناميكي في بعض الصناعات مثل صناعة الإلكترونيات. في هذا النوع من النظم، يعتمد اختيار الآلات وعمليات التصنيع على نوعية الطلبات خلال فترات زمنية محددة. لذلك يتميز كل طلب بتشكيلة معينة من المنتجات قد تتطلب تغييرًا في النظام الصناعي بناءً على وجود الآلات أو عوامل اقتصادية أخرى. يتطلب الأمر دراسة نظام التصنيع واعتبار العوامل المصاحبة لإعادة تشكيلة بما يتناسب مع الطلبات الجديدة. درس تشن (١٩٩٩م) هذه المسألة واقترح طريقة تقريبية للحل الأمثل تعتمد على تقسيم الأنموذج إلى عدد من الإجراءات. تم الوصول في هذا البحث إلى الحل الأمثل باستخدام الأنموذج الرياضي وكذلك تم اقتراح أنموذج جيني لحل المسألة، كما تم تقييم النتائج ومقارنتها مع النتائج السابقة باستخدام أمثلة رقمية.