International Journal of Industrial Engineering, 15(1), 62-72, 2008.

Performance Analysis of Flowshop Scheduling Using Genetic Algorithm Enhanced With Simulation

Ibrahim Al Kattan[1] and Rajashekar Maragoud[2]

[1]Director of Graduate Program
Engineering Systems Management,
American University of Sharjah, UAE

[2]Mechanical Engineering Department,
University of Massachusetts,
Dartmouth, MA 02747 USA

This paper uses the genetic algorithm combined with simulation techniques to evaluate the performance of flowshop scheduling. The problem of scheduling is a NP combinatorial optimization problem. The objective of the study is to develop robust scheduling by using Genetic Algorithms (GA) to obtain good solutions may be close to optimum sequence with minimum cycle time or make span time for a flow shop problem. The next step is to use simulation to analyze the performance of selected sequences to achieve better resource utilization. In this research, newly hybrid genetic algorithm is developed using C-programming language for four different methods to solve flow shop scheduling problems. The proposed methods are implemented on a number of tested problems and compared with exact solutions on smaller scale problems. The alternative sequence obtained here is further analyzed by simulating the production model using ARENA software.

Significance: The proposed hybrid CV-GA method on the flow shop problem showed that a better solution is obtained. The alternative sequence obtained here is further analyzed by simulating the production model using ARENA. Using simulation helps to improve the resource utilization and performance analysis.

Keywords: Flow shop problem; perfprmance analysis; hybrid scheduling algorithms; genetic algorithms, Simulation

(*Received 10 September 2005; Accepted in revised form 12 March 2007*)

1. INTRODUCTION

Scheduling is a decision making process to arrange activities in order to meet certain objectives and resource optimizations to perform tasks with a given set of criteria and constraints according to Sule, 1997. Resources can be machines, tools, materials, money, and labor. A flow shop scheduling problem (FSSP) is a typical combinatorial NP hard problem that accommodates technological constraint and is comprised of independent jobs with fixed order. A flow shop scheduling can be defined as scheduling the available jobs to be processed in a particular sequence or order on all machines to complete the given task satisfying the technological processing constraints. By scheduling, the jobs are given a priority to follow on machines in order to complete the given process with the main objective of optimizing the performance criteria. The sequence of the jobs that meets the required objective or a performance measure is termed as an optimum sequence. Recently much research has been done to obtain this optimum schedule. The flow shop problems in which the processing time of each job on each machine is given are known as deterministic flow shop problems. Thus the performance measure or the main objective of optimality for these deterministic flow shops is minimization of the total processing time of all jobs. If the jobs have the same sequence on each machine, such type of flow shop problems are called as permutation flow shop problems. This flow shop scheduling became more popular due to the high use of facility design in arranging workstations and material handling equipment. Theoretically, for an 'n' job on 'm' machine permutation flow shop-scheduling problems there are (n!) possible sequences. If the number of jobs 'n' is large, it is very difficult to obtain optimum solution from such a high number of sequence permutations. Thus a flow shop problem is considered as a NP-hard problem, this means the complexity or the computational costs of these problems increases exponentially with the problem size. Hence, for these nonlinear tasks, an approximate or heuristic algorithm can be a good choice to obtain efficient solutions. Many researchers have proposed heuristic methods to solve these NP-hard problems according to Baker,

1974, Lourenco and Helena, 2002. In this research the standard scheduling assumptions are used to solve typical flow shop problems.

2. LITERATURE SURVEY

The primary goal of this research is to develop and design certain rules that lead to the optimal or the closest optimal schedules satisfying the constraints with predefined criteria. A detailed theoretical analysis of the scheduling problem can be found in Garey and Johnson, 1079 , well known examples of the traditional artificial intelligent (AI) approaches were described in Balas, 1969, general coverage of the subject can be found in Muth and Thompson, 1963 and French,1982; less traditional AI approaches are described in Ow and Smith, 1988. A common NP scheduling problem in a manufacturing environment is Flow-shop problem. Johnson, 1954 was the first to propose an optimization problem with respect to the minimum makespan time for n jobs on a two-machine flow-shop-scheduling problem. Later Garey Johnson and Sethi, 1976 and Rinnooy Kan, 1976 proved that the flow-shop problems are NP-hard and therefore a practical solution can be only solved by approximation methods. A two-phase heuristic was proposed by Widmer and Hertz, 1989, the first phase is to construct the initial schedule and in the second phase tabu search techniques are used to improve the schedule. Taillard, 1990 in his paper presented comparative study on methods to solve the flow shop problem including an improved version of the heuristic proposed by Nawaz, Enscore and Ham, 1983 and a tabu search heuristic method.

Reeves, 1995 applied genetic algorithm successfully to solve traditional flow shop problem. Davis, 1985 was the first to suggest the use of genetic algorithms for scheduling using preference list based representation. For an n-job m-machine problem, a chromosome is formed of m subchromosomes, each for one machine. Each sub chromosome is a string of symbols with length of n, and each symbol identifies as operation that has to be processed on the relevant machine. Davis proposed an order crossover (OX) operator to his string encoding. OX selects the substring from a parent at random and produces proto-child by copying the substring exactly maintaining the same positions as they are in parent. Symbols that appear in the first parent are deleted from the second parent. The remaining symbols from the second parent are filled up in the child from left to right according to the order of the sequence generating an offspring. Syswerda, 1991, later proposed order-based crossover with slight variation of position-based crossover. Here the order of the position of symbols is imposed from one parent to the other. Bagchi, 1991, in his work concentrated on the problem-specific information part for the indirect representation of the schedules. Bagchi later concluded that the performance of the genetic algorithm could be enhanced by including more problem-specific information in the indirect method. Boctor, 1994 tried the simulated annealing approach using genetic algorithms for a scheduling project. The representation was a sequence-based hybrid, the genetic algorithm generates the sequences then the sequences were heuristically scheduled and this approach with a random precedence based shuffle performed better than the genetic algorithm for the traveling salesman problem. Husbands,2002 found that there is no difference between the sequence-based and scheduling problems. Syswerda, 1990, was the first to combine a genetic algorithm sequencer with a deterministic scheduler with special order based operators and later Hillard, 1988 implemented a classifier system to improve heuristics for determining the sequence of activities that should be sent to a scheduler, 1993. The local optimization is further accelerated and the quality of algorithms is further improved by the inclusion of heuristics to the genetic algorithms. The most common hybridization algorithms are obtained by combining the genetic algorithm with local search. In this hybrid method, during its learning the offspring passes on its characteristics to the future offspring through a common crossover, 1993. Giffler and Thompson, 1960 provide an algorithm that combines genetic algorithm with a heuristic rule in order to resolve the precedence constraints and to convert each offspring into an active schedule. Dorndorf and Persch, 1995, in their work used the genetic algorithms to evolve those chromosomes to find out a better sequence of machines for the shifting bottleneck heuristic, 2001. These recent successes of genetic algorithms (GAs) in optimization problems especially on scheduling problems encourage more research on this area. Multiple objective genetic algorithms (MOGAs) were also used to solve manufacturing problems involving a large number of variables. Scheduling with multiple objectives uses one single function that aims to reduce the total processing time to an extent. A MOGA technique divides the population into sub-populations, and allows each to optimize one objective, whilst information is exchanged between the sub-populations to allow the improvements to be made to solutions regarding the other objectives, 1999.

3. GENETIC ALGORITHM METHOD

Genetic Algorithm starts with an initial population, a set of strings or chromosomes. The population (set of chromosomes) that has high fitness values is combined and the stronger traits of each individual are exchanged (Crossover) and the weaker traits are neglected. Thus, produces children with stronger survival rate. As the GA is iterative this exchange of useful information among the chromosomes repeats until the stopping criterion is reached or until the population converges to obtain close to optimum solution. Sometimes there may be stagnation because the population may converge to a local optima, this can be avoided by introducing a mutation operator. Mutation operator changes the gene positions drastically at random and diversifies the population. This result will be used to introduce new population to the already existing

population. Thus, increases the probability of finding close to optimum solution according to Rajashekar, 2002. In early research Genetic Algorithms were represented as binary strings in most of the applications. For example '00110' represents an individual chromosome. Each chromosome consists of bits '1' and '0' and these bits are generally referred as genes, using the concept of bin-based representation, according Gen and Cheg 2001. The positions of these genes determine the value of the chromosome. The solution of a given problem is encoded into chromosomes and the information as genes. A fitness function is defined according to the criteria of best fit heuristic using bin-based representation. The requirement and the GA are operated with the objective of maximization of the fitness value in this case is to minimize the cycle time.

The most common form of initial population selection is random method. The numbers '1' and '0' are generated at random by a random number generator. The numbers thus generated are stored as chromosomes forming a population. GA operates on this population to converge on to an optimum solution or a solution closer to optimum. In recent years a number of 'greedy' heuristics were suggested in order to reduce the computational effort of the GA thereby increasing its efficiency. The solutions obtained from the heuristics are supplied to GA as initial seed. A good heuristic method can help a GA reduce its computational effort to a large extent. These GA's could be called as Hybrid Genetic Algorithms. The next step is to evaluate the fitness value of each chromosome. The fitness value is obtained from the fitness of performance criteria which is in this case the minimum cycle time of a batch production. The seeds generated by different methods helps the algorithm to start with population having better fitness values, which means, upper bound value of minimum cycle time thus resulting in faster rate of convergence toward a set of the possible good solutions. It is clear that only the individuals or strings with high fitness values are carried out on to the next generation and the other individuals in the population are discarded. The GA operates with the main objective is to achieve the best possible of fitness function. The individuals, which have reasonably good fitness value, are selected as parents. The parents determine the offspring characteristics hence the parents that have good traits are combined to get a fitter offspring. In the literature, many methods are proposed for parent selection like roulette wheel method and one-step selection method. The most common of them is roulette wheel selection method. In this method the individuals which are fitter were given a higher probability for selection, the individuals that are less fit are given lower probability. Thus an individual with high fitness has more chance to get selected for recombination and pass their good traits to next generation. But this roulette wheel method has many more steps; instead of this one-step selection method was proposed. In one step method the population is listed according to their fitness values. This means the fitness values are sorted and the best-fit individual is listed at first and the least fit is listed last. The parents are selected at random form the first few individuals in the sorted list. Thus fitter parents are selected for recombination viz., crossover and mutation in order to reproduce a fitter offspring. In this research one-step selection method is used to select parents from the given initial seed according to Rajashekar, 2002.

Crossover is the most important operator in GA by which the good traits of the parents are passed over to the child. Several crossover operators have been proposed for different problems. The most common crossover is one-point crossover. In one-point crossover, a crossover point is chosen at random. At this point the two parent chromosomes are divided and genes from one parent are exchanged with the genes from the second parent to form two new off springs, as swap mutation. Mutation is the process where the offspring has almost an identical chain as the parent except few genes that are changed (mutated) in order to prevent the stagnation of the population. In this operation two genes are selected at random and their positions are swapped. This is commonly known as swap mutation. Correction operator is used only when the chromosome loses its property and is not under the constraints after the recombination operations. This is a correction factor not required for all the applications. Replacement operator is a number suggested to replaces the old population when one generation is complete. In GA's where the entire current population or at least some of the current population is replaced by the new generation, the results were not encouraging. This is because entire replacement or random replacement of the population may result in the loss of useful combinations. In order to tackle this concept of elitism has been introduced. Elitism addressed this problem by always carrying fittest individuals to the next generation. This made the convergence easy but the computational effort is too high. Kim and Lee, 1994 suggested one replacement in one-generation concept and proved in their work that this method is much efficient compared to other methods. Hence, in this research one-to-one replacement method is used. In this method the replacement operator replaces the worst individual with the new one recombination operation if the offspring has unique value and fitter than the worst individual in the existing population. This replacement is repeated until there is no improvement in the fitness value, assuming close to optimum solution is achieved.

In this research, the GA is adapted to suit a flow shop problem. In a general flow shop problem there are 'n 'jobs and 'm' machine with the same sequence or the schedule of the jobs on all the machines. Here the individual jobs are considered as genes and the schedule formed by the jobs is considered as a chromosome. The makespan time or total completion time of the process is the fitness value for the chromosome. For Example: If there are 'n=5' jobs in a flow shop problem and the sequence followed by the jobs on all machine to complete the task, which is the total completion time taken is '100' then the chromosome is represented with fitness value as 100. The fitness value, which is the total completion time for a flow shop problem can be calculated for each chromosome, according to Rajashekar, 2002. The fitness value for the seed, in this case which is the optimum schedule represent the total process completion time (makespan time) taken by the given number of jobs in the schedule on all the machines to complete the process or task. A C program is written to calculate this fitness value or makespan time of each schedule and sort them. The processing times of jobs on

each machine are summed up based on the series defined in the schedule. The completion time of the last job on the last machine gives the total makespan time of the process or task. A Gantt chart displays the processing times of jobs on all machines following the above rules and assumptions. Form the chart the completion times of jobs on each machine can be observed and thus the idle times and the behavior patterns can be clearly seen. A Gantt chart can be defined as a line or block chart, which is used to display the scheduling information of jobs on machines. The x-axis of this chart represents the processing time and the y-axis represents the machines. It helps us in visualizing the sequence of jobs scheduled on each machine, the starting time for each job and the completion time for each. The makespan time of the schedule is shown by the completion time of the last job on last machine.

The application of a simple genetic algorithm may give some results, however to obtain a better solution, modifications have to be incorporated. Heuristic method could provide an effective solution to a hard flow shop problem and hence are used to complement the genetic algorithm. A hybrid approach is a method where the genetic algorithm performs the global exploration while the heuristic method performs the local exploitation around the chromosome. Hybridization can be performed either by incorporating heuristics into the initial pool of the genetic algorithm or by incorporating with the crossover operator's in order to improve the offspring by a quick local optimization. In a typical GA the initial seed is generated randomly. If heuristic rules are used to generate the initial pool instead of random method, the GA starts with population having higher fitness values. Thus, resulting in a faster rate of convergence compared to general GA methods. Considering the fact that GA method may allows diversity of the breeding process to escape a local optimum. The program calculates the fitness value (minimum cycle time) for all sequences at each stage. Hence, using the program to sort them, drop the extremely bad combinations and put the set with good fitness value into the pool. Therefore, proposed method keeps in the pool all populations with high fitness value so it leads to the convergence in the size of possible solutions. These types of hybridization methods are proposed in the next section and experiments are conducted to show their performance, according to Rajashekar, 2002.

4. PROPOSED HYBRID GENETIC ALGORITHM

In this research, new hybrid genetic methods are developed on the basis of generating initial seed. Four methods were developed to generate the seeding or initial population for the genetic approach. This seeding helps the algorithm to start with population having high fitness values, thus resulting in faster rate of convergence and increase the control over the process. These methods follow different rules to obtain individuals (sequences) with good fitness values. The individuals thus obtained are subjected to crossover and mutation in order to converge to an optimum sequence. A fifth method, which is the exact solution method, is developed to compare these results with the exact solutions for small value of 'n' jobs.

4.1 Method 1- Random Generation of Initial Seed

A computer program is written to generate the initial sequences randomly and to provide the genetic algorithm with the initial seed. The purpose of this initiative is to assign the bound. The fitness values are total process completion times of the generated population seed. The sequence is considered as the chromosome having the jobs as the individual elements (genes). The fitness value of each chromosome is calculated.

1. ***Rank the population****:* Rank the seed by sorting in ascending order of the makespan times that represent the fitness values. Hence, the first individuals in the list represent the elite or the best fit schedules in the seed and the last one represents the least fit individual.
2. ***Selection:*** Select a pair of schedules (chromosomes) at random from the above-sorted list. The sorting of the population is essential to select the pair of schedule (chromosomes) to add them in the pool. These selected individuals are crossed to obtain new offspring.
3. ***Crossover operator:*** A crossover point is chosen at random. The selected two parents are crossed at this point and the genes are exchanged. Two new offspring (child) are formed from the parents.
4. ***Correction Operator:*** According to the flow shop constraint, no job occurs more than once on a single machine in other words duplicity of the jobs is not allowed. Hence, duplicate jobs or the genes that occur more than once are replaced with the missing jobs in each offspring. Thus the schedule is strictly maintained to suit a flow shop-scheduling problem. Fitness values are calculated for each resulting child using makespan value.
5. ***Mutation operator:*** The next operation performed is mutation, here two positions of the individual genes in each chromosome are selected at random and their values are swapped. This results in the formation of another new offspring. Fitness value of the offspring is calculated.
6. ***Replacement Operator:*** The child formed from these operators is checked for fitness. If the fitness is better than the least fit individual in the initial seed and the fitness value is not the same as the one at the first population, the child replaces the least fit. If the child's fitness is no better than the least fit it is added up to the initial seed. By calculating the fitness value, the minimum cycle time; sorting them; drop the extremely bad fitness value; and keep all the rest into the pool. By this approach the initial seeds were generated and controlled. This replacement occurs once in one generation. Thus, the operators are repeated for a number of generations until an optimum fitness solution is found.

And also the program checks for the pre-defined convergence condition. The pre-defined condition is the number of generations the program has to be executed to obtain optimum solution/solutions. These solutions are stored and compared with the results from the other proposed methods.

4.2 Method 2- Initial Population by Grouping based on Co-Variance Factor

In this method the mean and standard deviations of the processing times of each job on all machines is calculated and the co-variance for each job is estimated. This is done by grouping the jobs in all possible combinations.

1. Calculate the mean (μ) of all the processing times of each job on all machines.
2. Similarly calculate the Standard Deviation (σ) of all the processing times of each job on all machines.

$$\mu_i = \frac{\sum_{j=1}^{m} P_{ij}}{m} \qquad \ldots (1)$$

$$\sigma_i = \sqrt{\frac{\sum_{j=1}^{m} (P_{ij} - \mu_i)^2}{m-1}} \qquad \ldots (2)$$

P_{ij} - Processing time of i th job on j th machine
n - total number of jobs available.
m-total number of machines.
μ_i- mean of the processing times of i^{th} job.

3. Calculate the Co-variance Factor for each job on all given machines.

As the size of the problem increases the number of combinations of sequences increase, hence the complexity of the problem.

4. The GA may result in a local convergence of all its population at some point instead at the exact solution. Hence the operators may be further modified or new operators may be added to
avoid this early convergence in order to explore the search further. The risk increases with an increase of problem size.

5.

$$CV_i = \frac{\sigma_i}{\mu_i} \qquad \ldots (3)$$

The initial pool generated by the heuristics for the GA implementation was limited in this research for larger problems. Hence the heuristics can be further modified to incorporate more initial pool and thus help in increasing the search space for the GA.

6. In the simulation, the modifications are made by increasing the resource capacity further modifications such as split work orders, reducing the customer delivery quantities, increasing the crew size etc can be made. However, this is in the discretion of the production manager, who takes decisions based on the nature of the production process desired and the nature of the company.
7. Sort the machines based on co-variance factor list
8. Divide the CV factor list into groups by dividing with the chosen number 'D'. Where D < n.
9. Number of jobs in each groups, g = n/D.
10. Each group is arranged in series to include all the jobs, thus forming a schedule or sequence for the jobs to follow on all workstations.
11. There is D! Possible permutations to arrange 'D' number of groups in series.

Thus D! The number of schedule or sequences of the jobs are generated. Since D < n, as a result, the original problem converted into aggregated problem with much smaller permutation numbers (n! << D!). The fitness value of each sequence, which is the total completion time, is calculated for the given processing times. Thus all the sequences obtained in the above steps combine to form an initial population seed for the genetic algorithm approach. This initial seed is fed to the GA method developed, which has crossover, mutation and replacement operators. Thus following the steps discussed in section 4.1, close to optimum solutions are obtained for this proposed method.

4.3 Method 3- Split Method of Generating Initial Seed

The proposed method generates the initial population seed and evaluate each individual's fitness as indicated in the following steps:

Step 1: Select any integer value 'K' with the range from 1 – 'm', where 'm' denotes to the total number of machines and 'K' is the number of machine in each cell. Divide the total number of machines 'm' with this number ' K' to obtain the number of cells or groups of machines to be processed.

For example: If the total number of machines 'm' is 6 and the value of 'K' selected is 1. Then 'm' is divided into six cells with one machine in each. If 'K' is 2 the then six machines divided into 3 cells. If there are five machines then (1 and 2 in first cell and 3, 4, and 5 in second cell and the other combination is 1, 2, and 3 in first cell and 4, and 5 in the second cell). By this all combinations were developed and the cycle time were calculated and sorted.

Step 2: In the second operation, calculate the averages of the processing times of all jobs for each cell. Use the Shortest Processing Time (SPT) method on all the cells. Let 'KS1' be the number of sequences or schedules is obtained.

Step 3: In the third operation, Longest Processing Time (LPT) method is used on all the groups to obtain again 'KS2' number of different schedules.

Step 4: In this step, Johnson's algorithm method is used to generate schedules for a group of two machines concept A and B machine. The groups are now copied into set 'G1' and the order of the groups are reversed and stored in set 'G2'. Considering different pair combination of the machine from the two given sets in order and using the Johnson's algorithm on each pair, different schedules are obtained. The cycle time calculated and sorted.

Step 5: In the final step, the number of machines 'm' is divided into two equal halves (A, B). If the number is odd then divide as (m+1)/2. The averages of the processing times of two or more machines or workstations are calculated in both groups 'A' and 'B'. Then add the middle machine to each side "group". Determine the averages of the processing times of all the jobs on each machine and repeat this until 'm' number of averages of the processing times is obtained. Using Johnson's algorithm on different combination pairs selected from both groups, a number of schedules are obtained.

Thus all the sequences obtained in the above steps combine to form an initial population seed for the genetic algorithm approach.

4.4 Method 4-Generating the Seed with the Hybrid Method

Sequences are generated as various job order combinations. Combinations are obtained by the summation of the processing time of jobs on each machine and evaluating them by using Shortest processing time (SPT), Longest processing time (LPT) and Johnson's algorithm methods [2].

For Example: For a '5' machines or workstations and 'n' jobs problem the Hybrid GA method can be explained as follows.

1. Sequence 1 can be obtained by arranging the processing times of jobs on the first machine using the SPT method.
2. Sequence 2 can be obtained by using the SPT method on the summation result of the processing times of jobs on machine 1 and machine 2.
3. Similarly Sequences 3, 4 and 5 can be obtained by using SPT method on the summation of processing times of jobs on machine 1, 2 & 3; machines 1, 2, 3 &4 and machines1, 2, 3, 4 & 5 respectively.
4. Sequence 6 is obtained by arranging the processing times of jobs on the machine 5 using LPT method.
5. Sequence 7 can be obtained by the summation of the processing times of jobs on machine 5 and machine 4 and arranging them according to LPT method.
6. Similarly Sequences 8, 9 and 10 can be obtained by using LPT method on the summation of processing times of jobs on machine 5, 4 and 3; machines 5, 4, 3 and 2 and machines 5, 4, 3, 2 &1 respectively.
7. Sequences generated using Johnson's Algorithm: Johnson's algorithm can be used to generate an optimum schedule only for n jobs on two machines. In the case of more than two machines, the machines are divided to form a set of two fictitious machines (A, B) for all possible combinations.

In the first case, the processing times of jobs on first machine is considered as that of machine 'A' and summing up the processing times of jobs on the remaining machines as machine 'B'. Thus by using the Johnson's algorithm on machine 'A' and machine 'B' the first sequence can be generated. By using Johnson's algorithm, a number of sequences are obtained. Secondly, the processing times of jobs on each machine starting to form machine 'A', the remaining machines are stored as machine 'B' one at a time. Thus a number of possible combinations of machine 'A' and 'B' are obtained. Using Johnson's algorithm on all the generated machines A's and B's, a number of sequences are obtained.

Thus all the sequences obtained in the above steps combine to form an initial population seed for the genetic algorithm approach. This initial seed is fed to the GA method developed, which has crossover, mutation and replacement operators. Thus following the steps discussed in method 1, close to the optimum solutions may be obtained for this proposed method.

4.5 Method 5 - Exact Solution Method

A program in C is developed to find an exact solution for a given problem. This program lists all the sequences for a given number of operations and calculates the minimum total completion time (MTCT) for each sequence. Sort the MTCT value, as fitness criteria to choose the optimum sequence. The optimality of each sequence is measured by its fitness value, which is determined on the basis of minimum of total completion time. A computer program in C written generates all the possible schedule combinations, calculate the MTCT and sort all possible sequences for jobs $n \leq 8$. This number is limited to eight because of memory constraints. It becomes computationally difficult to arrive at the exact solution. Hence paves the way for the genetic algorithm. The role of the proposed hybrid genetic algorithm is to explore the large search space of the possible sequence combinations for a given number of operations and converges towards an approximately close solution. As the size of the problem increases the number of combinations of sequences increase, hence the complexity of the problem.

1. The GA may result in a local convergence of all its population at some point instead at the exact solution. Hence the operators may be further modified or new operators may be added to avoid this early convergence in order to explore the search further. The risk increases with an increase of problem size.

2. The initial pool generated by the heuristics for the GA implementation was limited in this research for larger problems. Hence the heuristics can be further modified to incorporate more initial pool and thus help in increasing the search space for the GA.

3. In the simulation, the modifications are made by increasing the resource capacity further modifications such as split work orders, reducing the customer delivery quantities, increasing the crew size etc can be made. However, this is in the discretion of the production manager, who takes decisions based on the nature of the production process desired and the nature of the company.

5. APPLICATION AND ANALYSIS

The five methods elaborated in section 4 are put up for application in this chapter. Initially the four methods;1. Random Simple Genetic Algorithm method; 2. Split -GA method; 3. Co-Variance GA method and; 4. Hybrid GA method are considered for testing. The observations of these four methods are obtained for different processing time input generated as random numbers and their efficiency compared with the exact solution-method 5. Further these approaches are applied on two practical problems. The schedule that gives the minimum of all the makespan time resulted from the four methods is considered to be close to optimum solution for the given problem. A program in C is written to generate processing times at random for the analysis. Each set of processing times is generated with different range. In the following analysis the Exact Solution method is used to evaluate the performance of the proposed four methods. This method stands as a benchmark to compare the other methods. The optimum sequence obtained by the exact solution used to compare the four proposed methods with the measure of performance and sort them accordingly.

5.1 Testing the Proposed Methods Using Industrial Case

In this experiment, the processing times are taken from an Industrial problem with '31' parts and operating on '12' workstations. This problem is analyzed before by Kattan et al, 2003. In their work as a case study to explain their proposed method. The authors reached a close to optimum solution by applying Petri Nets to group scheduling in Flexible Manufacturing systems. Considering this problem as benchmark for medium size flow shop scheduling problems, the four-proposed GA methods are implemented by running the program. The four GA methods are implemented for three numbers of trials. The first trial GA is operated for '100' generations, in the second trial GA is operated for '150' generations and in the third trail GA is generated for '250' times. The best solution is obtained by CV-GA method with cycle completion time of "177"after 560 generations as shown in Table 1.

Table 1 Comparison of the Four Proposed Methods

Method	BOCT	BOCT (No. of Occurrences)		
		Trial 1	Trial 2	Trial 3
1. Random Simple GA Method	192	207(3)	195(1)	192(1)
2. Co-Variance GA Method	193	202(1)	200(1)	177(1)
3. Split GA Method	197	206(1)	200(1)	197(1)
4. Hybrid GA Method	193	194(1)	194(1)	193(1)

From the above observations, it can be concluded that after the first trial the best optimum completion time, BOCT. The optimum solution of all proposed methods is obtained from Hybrid GA method, which is '194' units. This BOCT is better than the solution obtained from the Petri Net approach, which is '196'. It is observed that after '560' generations of CV-GA method yielded close to optimum solution with BOCT of '177'. This solution could not be reached by any of the other GA methods. The schedule of the jobs that returned the best OCT is '{26 17 28 14 12 24 19 6 5 29 3 18 23 20 2 30 1 27 21 16 22 31 8 7 13 15 11 10 9 25 4}'. The best solution for job sequence or schedule obtained from the Petri Net approach for this problems is given as '{4 12 27 5 6 2 17 29 26 23 7 14 1 16 20 15 19 3 22 31 25 28 13 21 18 30 10 24 9 8 11}', according to Kattan et al, 2003. The BOCT for this schedule is calculated as '196'. Having more alternative schedules with the same

cycle time is good for any real problem. These alternatives could be analyzed in details by using simulation for another performance measures.

6. SIMULATION OF AN INDUSTRIAL CASE

Simulation helps in taking right decisions at appropriate time of production by accurately predicting the production bottlenecks, work-in-process (WIP) inventory, resource utilization and feasible schedules to reduce the total cost. In addition, simulation offers flexibility for production planning and scheduling and also offers many advantages like animation and easy visualization of activities of all events through dynamic graphs, plots, tables etc. Schedules of the parts can be modified or the capacity constraints of the resources could be changed in order to reduce the queues and throughput times in the system. This problem is normally addressed by increasing the capacity of the resources or decreasing the number of transporters or their speed or even by reducing the input. A detailed statistical analysis available with simulation also helps the production manager to view in depth into observations of the operations and the resource constraints and also to examine the possible decision consequences using different scheduling methods for the given production planning problem.

6.1 The Industrial Case

In this research, simulation is used on one of the schedules obtained from section 5 for 31 parts on 12 workstations of an industrial problem. Parts are processed in a certain chosen optimum schedule on all the workstations. This simulation model is analyzed for different performance measures like WIP, queue length and machine utilization using ARENA simulation software version 7.0. The production manager can choose any performance measure as criteria to select a schedule based on the statistics obtained from the simulation. In this process, 31 parts arrive at the arrival station as a batch of 1000. These 31 parts operate on 12 workstations; Shaping center, Turning center, Drilling center, Sawing center, Plating center, Welding center, Milling center, Grinding center, Reaming and Boring center, Measuring center, Inspection center and Packing center. After being processed on all the 12 workstations the parts are sent to depart section, which is shipping, and it is the exit station for the process. The simulation is run for 10 replications and at the end of each replication the statistics are reviewed and a counter is introduced at the exit system that gives the number of parts outputted. The parts follow the given schedule on all the workstations, first observation is conducted on Sequence S1 = {26 17 28 14 12 24 19 6 5 29 3 18 23 20 2 30 1 27 21 16 22 31 8 7 13 15 11 10 9 25 4} and the results are reviewed. In the later observations, modifications are made on the resource constraints to maintain the machine utilization and the average WIP under practical conditions.

6.2 Observations of the Simulation Results

Simulation models are built for the production problem defined in section 6.1 following the defined sequence. The results obtained are analyzed and here the performance criteria like the queue length, machine utilization and WIP are considered for the analysis. All the simulations are run by considering the arrival batch size of the parts as 1000; this means all the parts arrive 1000 at a time. Maximum number of batches allowed is fixed at 2. The output contains the average work-in-process (WIP) after 10 replications and also gives the number of parts exited the system after the given stipulated simulation time. In the first observation the simulation model is run for 10 replications with no modifications and the results are studied. The values of the performance measures like average WIP, queue length and utilization of the workstations (resources) are analyzed. The results showed the Average WIP as 871 and the number of systems out of the production line as 227. It is also observed from the results that some of the resources like the plating center and shaping center have high machine utilization of 100% with queue lengths of 137 and 692 respectively. The sawing and drilling centers have 95.15% and 94.9% of machine utilization with 18 and 10 as their queue lengths. It can be concluded that high resource utilization can increase the profits and also results in customer deliveries on time. But in practical conditions this may lead to a lot of congestion in the form of long queues and slow throughput and also the risk of unprecedented machine break down rises.

To avoid such a loss because of high resource utilization, production managers usually choose to split work orders, reduce the customer delivery quantities, increase the crew size or add more resources to the existing ones. In this model it is also observed that the average WIP of the parts is very high at 871 and number of parts that exited the system is low at 227 after 2000 units of simulation run time. Hence to improve these measures capacity of the resources whose utilizations are high were increased proportional to their individual queue lengths. As a first step, only five resources viz., plating, shaping, sawing, drilling and turning centers are considered based on their high utilization percentage and queue length as shown in Table 2. The modifications of the resource capacity are shown in the 5^{th} column of the Table 2. Simulation is performed on the model built with new modified capacities and the results are noted.

Table 2 Modification: Increase of the Resource Capacity

Resource	Current Utilization %	Queue Length	Current Capacity	New Capacity
Plating Center	100	137	1	5
Shaping Center	100	692	1	6
Sawing Center	95.15	18	1	4
Drilling Center	94.9	10	1	3
Turning Center	69.14	2	1	2

The plating center resource without any modification shows a very large queue length with maximum resource utilization of 100%. After the modifications, the queue length is decreased considerably and the resource utilization is also reduced. The average plating center utilization reduced from 100% to 51.3%, which means that the load on the plating resource is decreased thus reducing the risks for maintenance and unnecessary jamming of the parts. It is also clear that as the capacity of resources is increased the plot show that the queue length is reduced much significantly from 137 to 2 parts. A similar plot is drawn for shaping center, which also showed a high utilization and very large queue length. The average WIP, that shows the number of parts available in the system at a given time is also plotted and compared. This means that at the end of the replication run time only 522 parts are present in the system for processing and the remaining are departed or shipped, more production in less time.

7. CONCLUSIONS

In this research, a study of scheduling problems and the methods developed to solve them were elaborated. New heuristics were designed and developed to aid the genetic algorithm to converge on close to optimum solutions for given flow shop problem. Three heuristics were developed which are Split Method, Co-variance factor Method and the Hybrid GA Method of Generating Initial Seed. These heuristics are used to generate initial seed for GA and later the GA with the described operators is implemented to obtain best solutions. Later the results obtained are compared with the generic GA whose initial population is generated at random and compared with the exact solution method for problems having smaller number of jobs. The exact solution method is developed using C program to generate all possible schedule combinations for given jobs, then calculate each makespan and sort them. This limitation is due to the memory constraints of the computer. Thus for smaller scale problems exact solution method is used as benchmark and the results obtained showed that almost in all the observations the heuristic based hybrid GA's converged to the exact solution much quicker than the Random simple GA method. Implementation of the proposed hybrid GA methods on this problem showed that a better solution is obtained by the proposed CV-GA method compared to all the benchmark heuristic methods. After comparing and analyzing the hybrid GA's with different benchmark problems, it could be concluded that the proposed hybrid GA method is promising.

8. REFERENCES

1. Bagchi, S., Uckum, S. et al., Exploring Problem-Specific Recombination Operators for Jobshop Scheduling. Proceedings of the Fifth International Conference on Genetic Algorithms, 1991.
2. Baker, K.R., Introduction to Sequencing and Scheduling, New York: John Wiley & Sons, 1974.
3. Balas, E., Machine Sequencing via Disjunctive Graphs: An Implicit Enumeration Algorithm, Operations Research, 17:941-957,1969.
4. Blackstone, Jr. J.H., Philip, D.T. and Hogg, G.L., A Sate of Art Survey of Dispatching Rules for Manufacturing Job Shop Operations, International Journal of Production Research, 20(1):27-45,1982.
5. Boctor, F. F., An Adaptation of the Simulated Annealing Algorithm for Solving Resource-Constrained Project Scheduling Problems. Document de Travail 94-48 Qu.bec, Canada, Groupe de Recherche en Gestion de la Logistique, 1994.
6. Chen W. T., and Hao Hu, Production scheduling for precast plants using a flow shop sequencing model, Journal of computing in Civil Engineering, July 2002.
7. Cheng, Runwei and Gen, M., Production Planning and Scheduling Using Genetic Algorithms, Computational Intelligence in Manufacturing Handbook, Edited by Jun Wang et al, Boca Raton: CRC Press LLC, 2001.
8. Davis, L., Job Shop Scheduling with Genetic Algorithms. Proceedings of an International Conference on Genetic Algorithms and their Applications, Pittsburgh, Lawrence Erlbaum Associates, 1985.
9. Dorndorf, U. and Persch, E., Evolution Based Learning in A Job Shop Scheduling Environment, Computer Operations Research., vol.22, no.1, pp.25-40, 1995.
10. French, S., Sequencing and Scheduling an Introduction to the Mathematics of the Job-Shop, Ellis Horwood, 1982.

11. Garey, M. and Johnson, D., Computers and Intractability: A guide to the theory of NP-Completeness, W.H.Freeman, 1979.
12. Garey, M.R., Johnson, D.S., and Sethi, R., The Complexity of Flowshop and Jobshop Scheduling, Mathematics of Operations Research, 1:117-129, 1976.
13. Gen, M. and Cheng, R., Genetic Algorithm & Engineering Design, John Wiley & Sons. Inc. 1997.
14. Gen, M. and Cheng, R., Genetic Algorithm & Engineering Optimization, John Wiley & Sons. Inc. 2000.
15. Giffler, B., and Thompson, G.L., Algorithms for Solving Production Scheduling Problems. Operations Research, Volume 8(4):487-503, 1960.
16. Hillard, M. R., Liepins, G. E., et al., Machine Learning Applications to Job Shop Scheduling. Proceedings of the AAAI-SIGMAN Workshop on Production Planning and Scheduling, 1988.
17. Husbands Philip, Genetic Algorithms for Scheduling, AISB Quarterly, No. 89, 1996. URL: http://citeseer.nj.nec.com/149983.html last viewed on June 2002.
18. Johnson, S.M., Optimal Two and Three-Stage Production Schedules with Setup Times Included. Naval Research Logistic Quarterly, 1:61-68, 1954.
19. Kattan, A. Ibrahim, Bolek Mikolajczak, and et al, "Minimizing cycle time and group scheduling, using Petri nets – A study of heuristics methods. Journal of Intelligent Manufacturing, Vol. 14 PP 107-121, 2003, Kluwer Academic Publisher Netherlands.
20. Kennedy, S., Five Ways to A Smarter Genetic Algorithm, AI Expert, pp. 35-38, 1993.
21. Kim, G.H. and George Lee, C.S., Evolutionary Approach to the Job-Shop Scheduling problem. IEEE Int. Conference Robotics and Automation. vol.1, pp.501-506, 1994.
22. Lourenco, Helena, R., Sevast'yanov's Algorithm for the Flow-Shop Scheduling Problem. URL: http://citeseer.nj.nec.com/216417.html. last viewed on May 2002.
23. Matthew Bartschi Wall, A Genetic Algorithm for Resource-Constrained Scheduling. Department of Mechanical Engineering in partial fulfillment of the requirements for the degree of Ph.D. in Mechanical Engineering ©1996 MIT.
24. Muth, J. and Thompson, G., Industrial Scheduling, Prentice Hall, 1963.
25. Nawaz, M., Enscore, E. E. Jr. and Ham, I., A Heuristic Algorithm for the m-Machine n-Job Flow-shop Sequencing Problem, OMEGA, International Journal of Management Science, 11(1), 91-95, 1983.
26. Ow, P. and Smith, S., Viewing Scheduling as an Opportunistic Problem Solving Process, Annals of Operations Research, 12, 1988.
27. Peter Ross, Hsiao-lan Fang and Dave Corne, Genetic Algorithms For Timetabling and Scheduling, Proceedings of Expert 94, Pretoria, South Africa, October 1994; Computer Society of South Africa
28. Rajashekar Maragoud, MS thesis, Design and Analysis Of Hybrid Genetic Algorithm Approach For Production Planning And Scheduling, university of Massachusetts Dartmouth, 2002.
29. Reeves, C. R., 1995, A Genetic Algorithm for Flow shop sequencing, Computers and Operations Research, 22, 5-13.
30. Rinnooy Kan, A.H.G., Machine Scheduling Problems: classification complexity and computations, Nijhoff, The Hague, 1976.
31. Shaw, J., Multiobjective Genetic Algorithms for Schedule Operation, Manufacturing Complexity Network Meeting, Warwick Manufacturing Group, 3rd Aug. 1999, pp. 8-12.
32. Sule, D.R., Industrial Scheduling, PWS Publishing Company, Boston, MA, 1997.
33. Syswerda, G., Schedule Optimization Using Genetic Algorithms. Chapter 21 of the Handbook of Genetic Algorithms, New York, New York, Van Nostrand Reinhold, 1991.
34. Syswerda, G., The Application of Genetic Algorithms to Resource Scheduling. Proceedings from the Fourth International Conference on Genetic Algorithms: 502-508, 1990.
35. Taillard, É., Some Efficient Heuristic Methods for the Flow Shop Sequencing Problem, European Journal of Operational Research, vol. 47, 65-74, 1990.
36. Widmer, M. and Hertz, A., A New Heuristic Method for the Flow Sequencing Problem, European Journal of Operational Research, vol. 41, 186-193, 1989.

BIOGRAPHICAL SKETCH

Ibrahim Al Kattan is the Director of Engineering Systems Management a graduate program, at American University of Sharjah in United Arab Emirates. He has a Ph.D. in Industrial and Manufacturing Engineering from Tennessee Technological University, an MS in Industrial Engineering and Management from Oklahoma State University, and another MS in Engineering Production and Management from the Birmingham University, England. Dr. Kattan was an associate with "Center of Excellence", Center for Manufacturing Research at Tennessee Technological University over ten years, 1986-1997. He was an active member at Advanced Technology and Manufacturing Center, Associate Professor of Mechanical Engineering at

University of Massachusetts for six years, 1998-2004. Dr. Kattan has over twenty five years of experience as a teacher, researcher, consultant in engineering management, simulation and modeling, quality management, and supply chain management in USA, UAE, UK and Iraq.

Rajashekar Maragoud is currently pursuing a Master of Science degree in Mechanical Engineering from the University of Massachusetts Dartmouth, 2002.