A Genetic Algorithm Application: Dynamic Re-configuration in Agile Manufacturing Systems

Subhash Wadhwa and Ashish Chopra

Department of Mechanical Engineering

Indian Institute of Technology

New Delhi 110019

INDIA

E-mail: swadhwa@mech.iitd.ernet.in

Abstract: Agile Manufacturing Systems are evolving to meet the challenges of an increasing time based competition in high variety environments. This paper is motivated by a study of an Indian Enterprise in the Telecom Sector where a conventional low variety company is seeking avenues for re-organizing itself towards agility. We propose a novel concept of *Dynamic Re-configuration with Integrated Scheduling (DRIS)* in manufacturing systems, to achieve greater agility. It is suggested that the inherent flexibility in many multi-skilled workforce based assembly systems allows dynamic re-configuration opportunities. However without effective decisions on when to re-configure, how much to re-configure and what order to schedule on a re-configured system, it is not possible to exploit this opportunity. We propose the use of a Genetic Algorithm (GA) based technique to obtain reasonably good schedules for such a dynamic re-configurable production system. The technique has been demonstrated using a simple model of a production system comprising multiple production lines, each capable of being re-configured into two or three independent production lines or re-combined into single lines, *dynamically*. The results indicate that the performance of the GA based technique is superior to the standard scheduling rules of "Shortest Processing Time" (SPT) and "Earliest Due Date" (EDD). The results also indicate that as the variability of demand on production system increases, GA based technique comes to better solutions for the integrated scheduling of dynamic re-configuration systems. It is proposed that DRIS approach can offer immense opportunities for the agile enterprise capabilities.

Professor **Subhash Wadhwa** (Eur. Ing., C. Eng.) received the Ph.D degree from the Department of Industrial Engineering, UCG, Ireland, while working on an ESPRIT project on Integration of Robots into Computer Integrated Manufacturing. He extensively contributed to the development of generalized simulators and expert systems to assist the evaluation of flexible assembly systems. After obtaining the Ph.D degree he worked at Digital Equipment Corporation, an US multinational company, in the area of Knowledge Engineering, and contributed to another ESPRIT project on Integrated Manufacturing Planning and Control Systems. He joined Indian Institute of Technology, New Delhi in 1990. He is currently Associate Professor. He has over 30 publications and communications in international journals and conferences and co-authored a chapter of a book edited by U. Rembold. His research interests and consultancy areas focus on FMS, decision systems, simulation, AI in Manufacturing, ERP and Logistics.

1. Introduction

Agile Manufacturing is an emerging concept in industry that aims at achieving flexibility and responsiveness to the changing market needs. We propose that dynamic re-configuration with integrated scheduling (DRIS) can offer immense opportunities for agile systems. Dynamic reconfiguring of production systems enables enterprises to respond to market in a more efficient manner and to obviate the need for other options such as capacity addition or worker retrenchment. For example a team of multi-skilled workers may be broken down into independent sub-teams, which may be assigned to different tasks, as and when needed. However such a dynamic re-configuration requires an integrated scheduling which is quite opportunistic in nature. It may thus be imperative for an agile enterprise to handle the demand variations by dynamically re-configuring its production systems and opportunistically re-scheduling them. However, the very capability of dynamic configuration of production systems makes finding a near optimal schedule be a vexed problem. The possible states of the system increase manifold thereby increasing the complexity to unmanageable proportions. Genetic algorithms have been theoretically and empirically proven to provide robust search in such complex spaces. Hence it has motivated us to study the performance of a Genetic Algorithm based technique for scheduling of DRIS systems, as compared with the conventional scheduling rules "Shortest Processing Time" (SPT) and "Earliest Due Date" (EDD). This paper is an attempt in this direction.

2. Application of Genetic Algorithms

Genetic algorithms are search algorithms based on the principles of natural selection / survival of the fittest to obtain reasonably good solutions to complex problems. These algorithms are not limited by restrictive assumptions about the search space (such as continuity, existence of derivatives etc.). Genetic Algorithms use a population of candidate solutions to conduct a robust search of the search space. The initial population of candidate solutions is generated randomly. Each such solution is then considered by assessing its value according to a fitness function. A new population of candidate solutions is then generated from the old population by applying the three operators of the Genetic Algorithm, viz., reproduction, crossover and mutation. Each such iteration is called a generation. The inherent power of Genetic Algorithms ensures that the successive generations of solutions are superior to their ancestors in terms of their fitness values.

Genetic Algorithms have been invented by Holland (1975) and used for a wide variety of problems such as machine learning [Booker et al, 1989], cellular manufacturing [Balakrishnan and Jog, 1995] , combinatorial optimization [Jog et al, 1991] and game playing [Axelrod 1987]. The application of GAs in scheduling was first introduced by Davis (1985). Liepins et al (1987) investigated the simplest scheduling problem of a static queue of jobs with specified due dates and run times without precedence constraints. Gupta et al (1993) studied a single machine model with an objective to minimizing flow time variance. Lee and Kim (1995) examine the performance of a parallel GA for a model in which earliness and tardiness penalties are allowed to be arbitrary and jobs share a common due date. Lee and Choi (1995) consider a more general model where the job due dates and early/tardy penalties are allowed to be arbitrary. Cheng *et al* (1995) consider a model of identical parallel machines where the objective is to minimize the maximum weighted absolute lateness about an unrestricted common due date. Jain et al (1997) developed an initial schedule by using genetic algorithms and addressed rescheduling in an environment of machine breakdowns, increased order priority, rush orders arrival and order cancellations. Goldberg (1989) provided an exhaustive study of genetic algorithms and of their various applications. This paper demonstrates the utility of Genetic Algorithms for integrated scheduling of the proposed novel domain of DRIS systems.

We apply genetic algorithm based technique to obtain reasonably good schedules under alternative scenarios of dynamic re-configuration. The re-configuration possibilities involve configuring two or three sub-lines out of any existing full production line (called breaking of a line) and combining them back into the full line (called Unbroken Production Line). The line can be broken in two modes, i.e. simultaneous or phased breakup with different logical implications for scheduling. This is explained in the next Sections. The different alternative approaches applied to GA are : "no break-ups of the production lines allowed", "simultaneous break-ups allowed" and "phased break-ups allowed". These are also compared with standard scheduling rules "Shortest Processing Time" and "Earliest Due Date" with no break-ups allowed.

Industrial Motivation

The principal author has been a consultant to the ABC company which is a fast growing, professional, Telecom industry in India. It is increasingly facing the variety challenge requiring agility. The customers are demanding smaller and smaller volumes of a variety of products in more and more timely manner. In the past the company has invested in large assembly lines to achieve high efficiency for large volumes (low variety) with few dedicated customers. The aim is to become more agile to deal with the new challenges, using the existing production capabilities.

The system analysis of the company indicates some interesting untapped potentials. For instance the workforce is virtually multi-skilled as the tasks involved are quite similar. In the past, training has been so imparted that any worker can contribute to any assigned task on the line. This was done to ensure that some absentees do not halt the lines. Another interesting potential results from the nature of the assembled products. Further the product design is such that the precedence constraints are relatively fewer across alternative assembled products. Infrastructure wise every station on the lines is equipped with the basic tooling required to assist all the assembly tasks. This was done in the past to ensure that a complete product line may be conveniently re-configured into a new line of appropriate size. All of this was done in the past with an overall assumption that the company will always have a few, large volume customers for whom dedicated product lines could be easily set up. Some of the assumptions have failed.

The emerging business scenario is however quite different than what was expected. There is a growing need for delivering small volumes of high variety products for a larger spectrum of end customers. Therefore the use of large lines to produce large volumes efficiently is no more a viable solution in the new scenario. Keeping in view the available potential flexibility (not used in large lines), our challenge is to re-configure the system. We propose a novel approach of dynamic re-configuration of production systems involving integrated scheduling (DRIS). We have developed a demonstrative DRIS model that highlights this approach. In our opinion the underlying concepts can play a significant role in Agile manufacturing systems. To deal with the scheduling complexity due to the dynamic re-configuration features in DRIS, we use the GA techniques to develop the schedules more effectively. The DRIS model is based on the discrete event perspective employing decision points. Similar to the decision points described by Wadhwa and Browne (1990), the decision points in DRIS model may be viewed as exploiting the flexibility of the re-configurable system (i.e. dynamic re-configuration).

The DRIS model uses an underlying concept of decision-information synchronization (DIS) at each of its decision points. It follows an overall framework of the GRAI macro reference model [Doumeingts et al, 1995] applied at the operational level. However instead of a periodic decision framework it is based on the event driven decisions where specific events offer system re-configuration opportunities. The DRIS model structure can easily be extended to deal with the DIS delays [Wadhwa and Bhagwat, 1998]. The latter will be useful for greater applicability in the Indian Industry where preference is for phased CIM developments at the operational levels. However for the purposes of this paper, we are motivated to demonstrate the DRIS model within a real time control environment involving no DIS delays.

Demonstration Model

The demonstration model comprises a production system which consists of M production lines and P products. The production lines have the following characteristics:

- Each production line can be broken down into two or three parts. Once the parts have been created, each part functions as an independent production line. At any time, all the parts can be combined together to form the original unbroken production line.
- A production line which was part of an original unbroken machine, cannot be broken down further.
- Each production line is capable of manufacturing all the products. This is also true for production lines obtained by breaking down original unbroken production lines.

Figure 1 shows the various possible states of a production line.

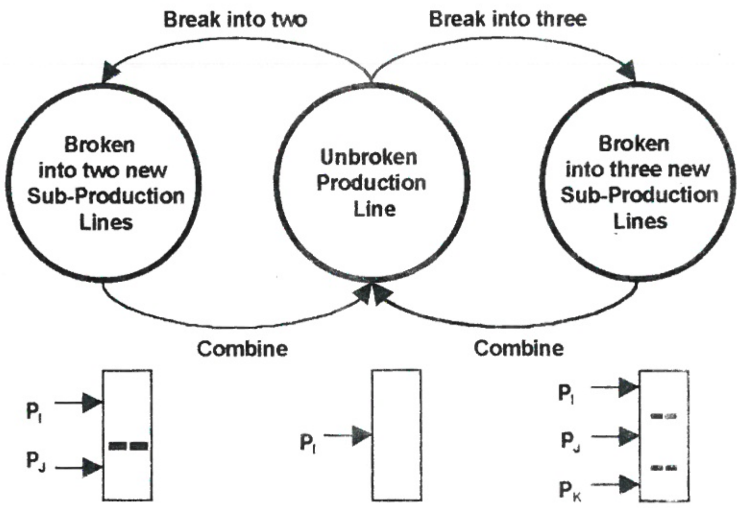


Figure 1. Possible Configurations of One of the Many Production Lines

Problem Scenario

The problem is to develop a near optimal schedule for the production of the given set of products on the production system, which can be dynamically re-configured, using a Genetic Algorithm based technique.

A production line is denoted by:

M[ORIGINAL , PARTNO,TOTAL]

Where:

ORIGINAL:	number of the original unbroken production line
PARTNO:	Part number of this production line out of the total existing parts.
TOTAL:	total number of parts which the original production line was broken into.

Thus M[2,1,3] indicates that this production line was obtained by breaking the 2nd original production line into 3 parts and this is the 1st part out of the three. An original production line will have PARTNO and TOTAL equal to zero. E. g. M[2,1,1] is the 2nd original unbroken production line .

Variable factors:

(a) Setup time:

Setup time means the time taken to ready the production line for the first time so that products can be put on it subsequently .

(b) Changeover time:

Changeover time is the time taken to changeover to the production of a different product than the previous one.

(c) Production Rate:

It is the rate at which units of a particular product are produced on a particular production line per unit time.

(d) Due Date:

It is the time by which the required quantity of a product should be ready.

(e) Demand:

It is the required number of units of a product to be produced.

Breakage:

Each production line can be broken into two or three parts. The production data for the new sub-production lines would be identical, i.e. they will have the same production rates for different products, same changeover times, etc.

The type of breakage can be :

(a) Simultaneous:

All the sub-production lines become available at the same instant after some time.

$S_S = [\, 0.2 + (TOTAL/3)*(0.5)\,]* S_I$

Where

S_S : Setup time of the new sub-production line

S_i : Setup time of the original production line

20% of original setup time is a fixed component while the rest is the variable component depending upon the number of parts which the production line is broken into.

E. g. let M[1,1,1] have Setup time = 100.

Then

If TOTAL =2 For M[1,1,2] and M[1,2,2]
Setup time = 53.

If TOTAL=3 For M[1,1,3],M[1,2,3] and M[1,3,3]
Setup time = 70.

(b) Phased:

The sub-production lines become available at different instants of time. For example, If the number of production lines = 2, then

TOTAL = 2

$S_1 = 1.2*S_i / 2$

$S_2 = 1.2*S_i$

Where

S_1 : Time after which first sub-production line becomes available

S_2 : Time after which second sub-production line becomes available

S_i: Setup time of the original production line

For example

if for M[1,1,1]	Setup time = 100
For M[1,1,2]	Setup time = 60
For M[1,2,2]	Setup time = 120.

TOTAL=3:

$S_1 = 1.2*S_i / 3$

$S_2 = 2*1.2*S_i / 3$

$S_3 = 1.2*S_i$

Where

S_1 : Time after which first sub-production line becomes available

S_2 : Time after which second sub-production line becomes available

S_3: Time after which third sub-production line becomes available

S_i : Setup time of the original production line

For example

if for M[1,1,1]	Setup time = 100,
For M[1,1,3]	Setup time = 40
For M[1,2,3]	Setup time = 80
For M[1,2,3]	Setup time = 120

The time for breakage acts as a penalty for changing the configuration of the system. It is modeled in such a way that it has a fixed component and a variable component to ensure different penalties for different types of breakups. The reason why to make a distinction between simultaneous and phased breakage is because of the different benefits and costs involved. While the cost for simultaneous breakage would be greater than phased breakage, the total time by which the entire capacity (arising out of breakage) is available for production is less with simultaneous breakage than with phased breakage. Each of these may be beneficial in different scenarios.

Production Rates:

For both simultaneous as well as phased breakage

$$P_s = 0.9 * P_i / TOTAL$$

Where

P_s = Production rate of a product on the new sub – production line.

P_i= Production rate of a product on the original production line.

Thus if for M[1,1,1]

P1	P2	P3
10	20	30

TOTAL = 2:

For M[1,1,2] and M[1,2,2]

P1	P2	P3
4.5	9	13.5

TOTAL=3:

For M[1,1,3], M[1,2,3] and M[1,3,3]

P1	P2	P3
3	6	9

The factor of 0.9 is for imposing a penalty for breakage.

Changeover Time:

For both simultaneous as well as phased breakage

$$C_s = C_i / TOTAL$$

Where

C_s : Changeover time for a product to another product on the new sub-production line.

C_i : Changeover time for a product to another product on the original production line

For example if for M[1,1,1] Changeover time from product 1 to product 2 is 100.

TOTAL =2:

For M[1,1,2] and M[1,2,2] Changeover time from product 1 to product 2 is 50.

TOTAL=3:

For M[1,1,3], M[1,2,3] and M[1,3,3] Changeover time from product 1 to product 2 is 33.

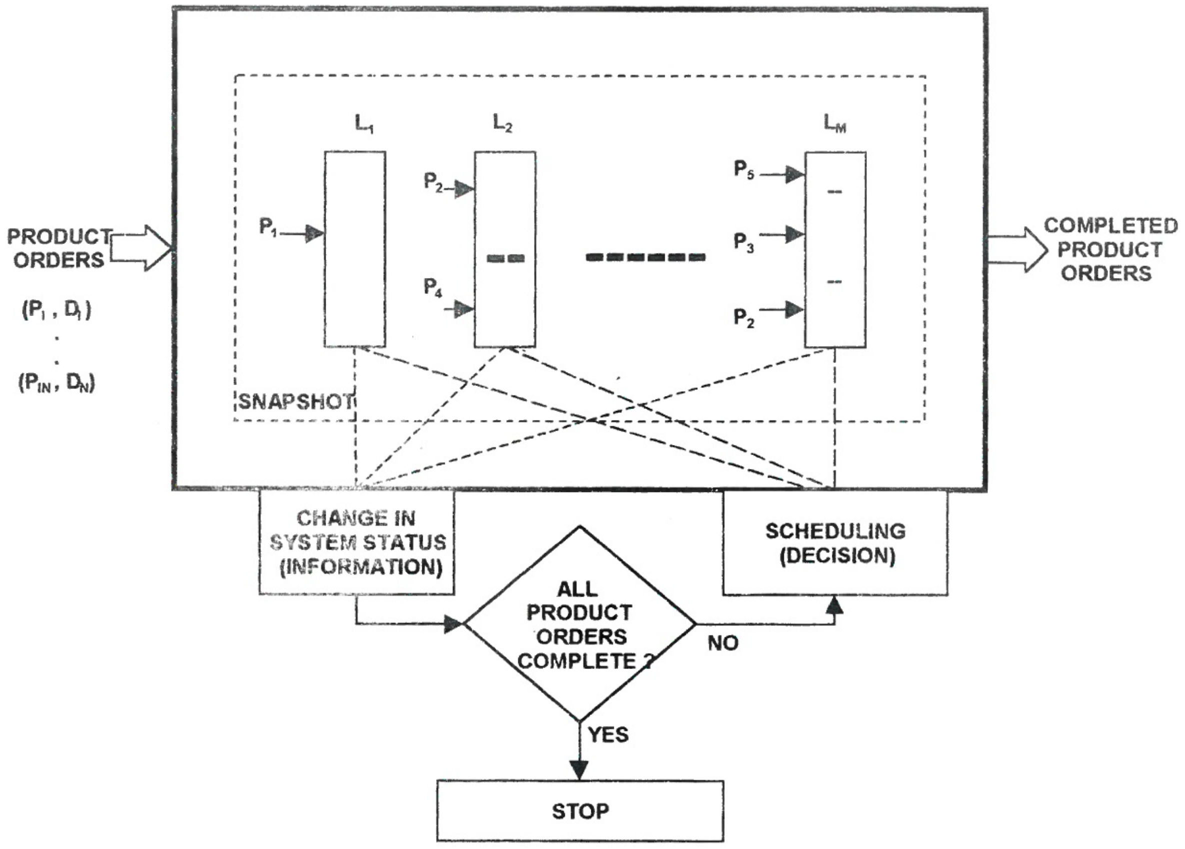


Figure 2. Schematic Representation of the Re-configurable Production System

Combining:

The sub-production lines can be combined together to form the original unbroken production line.

Combination time = (Setup time of original production line) / TOTAL

For example, if Setup time of M [1,1,1] is 100, then the time taken for combining M [1,1,2] and M[1,2,2] back into M[1,1,1] is 100/2=50.

Figure 2. shows the snapshot of the production system at a particular instant of time.

Genetic Algorithm Implementation

Now we explain the GA implementation within the DRIS model described above which reflects a typical industrial scenario.

(a) Schedule representation:

The most difficult part in a genetic algorithm formulation for scheduling is the schedule representation. A linked list is used for schedule representation. The use of a linked list is imposed by the dynamic re-configurable nature of the manufacturing system being modeled. Figure 3 shows the linked list representation of the schedule in the computer program.

All the decisions made on a machine are stored in the order in which they are made. The decisions can be:

- If it is an original machine, break it into two or three parts.
- If it is not an original unbroken machine, then combine all its parts.
- Put a particular product on the machine, no matter what the type of machine.

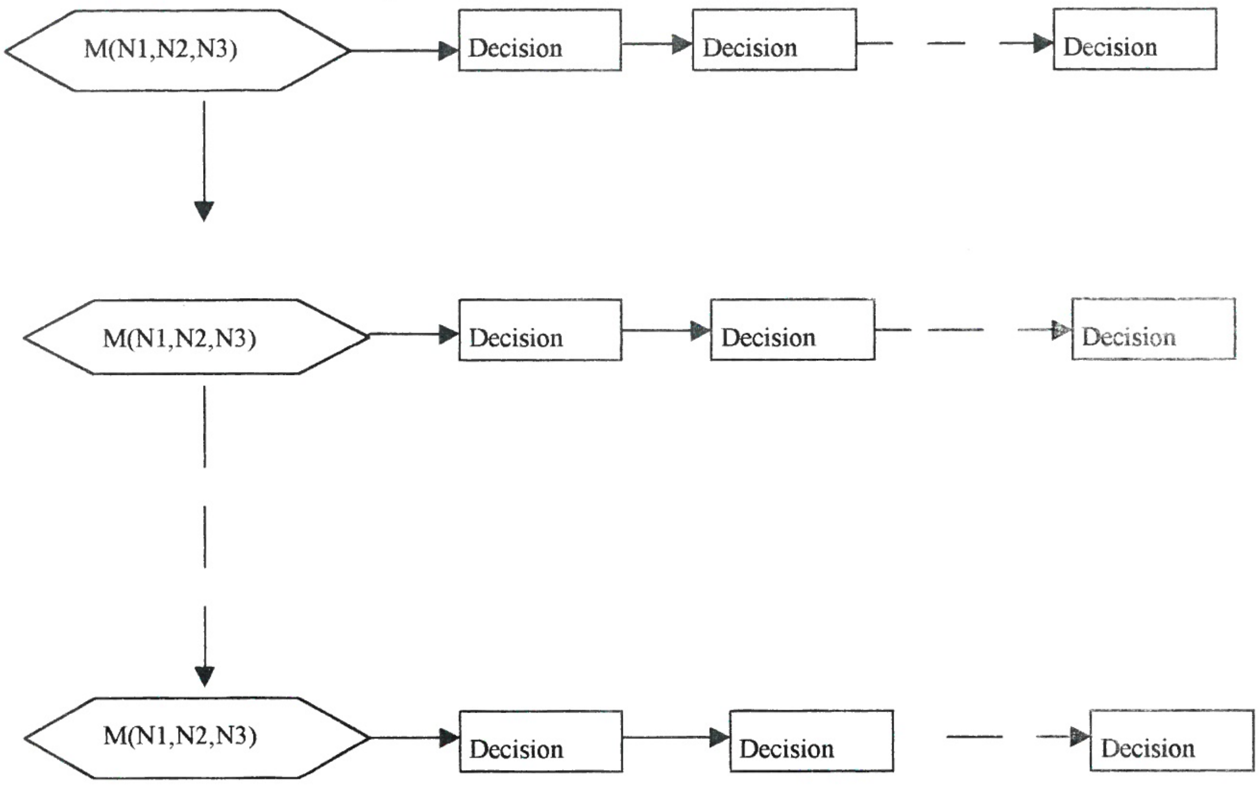


Figure 3. Schedule Representation

(b) Decision representation:

A decision is represented by the following notation.

P(N) :	Put product N on the production line.
B(N) :	Break the production line into N parts.
C :	Combine all the production lines into the original unbroken production line.
R :	Take a random decision during simulation of the schedule.

An example of a schedule for a production system of 3 original machines and 8 products is given below.

```
(1,1,1)  P(7)  P(8)  P(3)  P(5)  B(2)  R
(2,1,1)  P(7)  P(4)  B(3)  P(1)  B(3)  R
(3,1,1)  P(2)  B(3)  P(5)  B(3)  R     R
(1,2,2)  P(1)  P(1)  C
(1,1,2)  P(1)  P(2)  P(7)
(2,2,2)  P(4)  P(2)  C     P(7)  R
(2,1,2)  P(8)  P(7)  R
(3,3,3)  P(4)  P(8)  P(1)  R
(3,2,3)  P(6)  C     P(1)  R     R
(3,1,3)  P(6)  P(1)  P(7)  R     R
(2,3,3)  P(1)  P(5)  R     R
(2,2,3)  P(6)  P(8)  P(5)  C
(2,1,3)  P(1)  P(5)  R
(1,3,3)  P(5)  R     R     R
(1,2,3)  P(5)  R     R
(1,1,3)  P(5)  R     R
(3,2,2)  P(5)  R     R     R
(3,1,2)  P(5)  R     R
```

(c) Generation of initial population:

An initial population of schedules is generated randomly using simulation, with the number of schedules in the population equal to MAXPOP. Decisions are taken randomly whenever an event occurs.

(d) Genetic Algorithm Operators:

The genetic algorithm uses three operators viz., reproduction, crossover and mutation, on the existing population of schedules, to generate a new fitter population of schedules. Each of these processes is briefly described below.

Reproduction

Reproduction is a process in which individual schedules are probabilistically selected according to their fitness values. Schedules with a higher fitness value have a higher probability of contributing one or more copies in the next generation of schedules. The reproduction operator is implemented in the form of a biased roulette wheel where each current schedule in the population has a roulette wheel slot sized in proportion to its fitness. Each time a new off-spring is required, a simple spin of the weighted roulette wheel yields the reproduction candidate.

Crossover

Each pair of schedules selected from the previous population of schedules by the reproduction operator is called MATE 1 and MATE 2. These two schedules are then standardized into a common form and the new schedules are generated using a crossover operator. For standardization of the mates, the following standardization algorithm has been used:

Standardization algorithm:

1. Start with the first machine node (MN) in MATE 1

2. For each machine node (a machine node specifies the sequence of decisions to be taken on the particular distinct machine during simulation of the schedule) in MATE1 MN1, locate the machine node MN2 for this machine in MATE2.

3. Reposition MN2 in the same order as of MN1 in the order of machine nodes in MATE1. If MN2 does not exist, create a new machine node in the same order as of MN1 in MATE1.

4. Equate the number of decisions in MN1 and MN2 by adding dummy decision nodes to the machine node having less decisions. A dummy decision node specifies a random decision to be taken at the time of simulation of the schedule.

5. After going through all MN1, check for any further MN2 in MATE2 which are not present in MATE1. If any, create the corresponding machine nodes at the end of MATE1 in the same order as in MATE2. Add dummy decision nodes to these new MN1 equal to the number of decisions in the corresponding MN2 in MATE2.

6. Stop.

After MATE1 and MATE2 have been standardized into a common form, the following parameters are evaluated:

TOTALD :	total number of decision nodes in the two mating schedules. All the decisions in each schedule are indexed.
POSIT1 :	the index of the first decision node which is different in MATE1 and MATE2. It may have a value from 1 to TOTALD.
POSIT2 :	the index of the last decision node which is different in MATE1 and MATE2. It may have a value from 1 to TOTALD.
NOOFS :	the total number of decision nodes which are different in MATE1 and MATE2.
COEF :	the relative similarity of the two schedules MATE1 and MATE2.

COEF = NOOFS / TOTALD

The two schedules are crossed over with a probability PCROSS. If no crossover is to be done, MATE1 and MATE2 are copied as such into CHILD1 and CHILD2. Else if crossover is to be done, reduced surrogate crossover operator is applied. The probable crossing sites range from between the decision nodes with indexes POSIT1 and POSIT1+1 to after the decision node with index POSIT2. A crossing site JCROSS is selected uniformly at random between POSIT1 and POSIT2. Two new schedules CHILD1 and CHILD2 are generated by swapping the decision nodes with an index greater than JCROSS.

Mutation

Mutation performs a secondary role in the operation of genetic algorithms. Mutation is needed because even though reproduction and crossover effectively search and re-combine extant schedules, occasionally they may become overzealous and lose some potentially useful genetic material (decision nodes at certain places which under certain conditions may improve the overall schedule). Mutation operator protects against such an irrevocable loss. Adaptive mutation operator has been used instead of normal mutation. It bases disruption of a schedule on two factors- relative similarity of its two parent schedules and a mutation probability PMUTATION. The more similar the two parent schedules are, the more likely is mutation to occur.

ACTUAL MUTATION PROBABILITY (AMP) = COEF * PMUTATION.

Each decision node of CHILD1 and CHILD2 is mutated with a probability equal to AMP. If mutation is to be performed on the decision node, then a new randomly selected decision is stored in the decision node. After the three operators-reproduction, surrogate crossover and adaptive mutation have been applied, the final new schedules CHILD1 and CHILD2 are stored in the new population.

By applying these operators repeatedly on the previous population, a new population of MAXPOP individual new schedules is generated.

Figure 4 describes the application of the crossover and mutation operators as a flow chart.

SCHEDULE FITNESS

The schedules in the initial starting population are obtained by simulation and the evaluation variables are calculated thereof. The schedules obtained by reproduction, crossover and mutation are simulated.

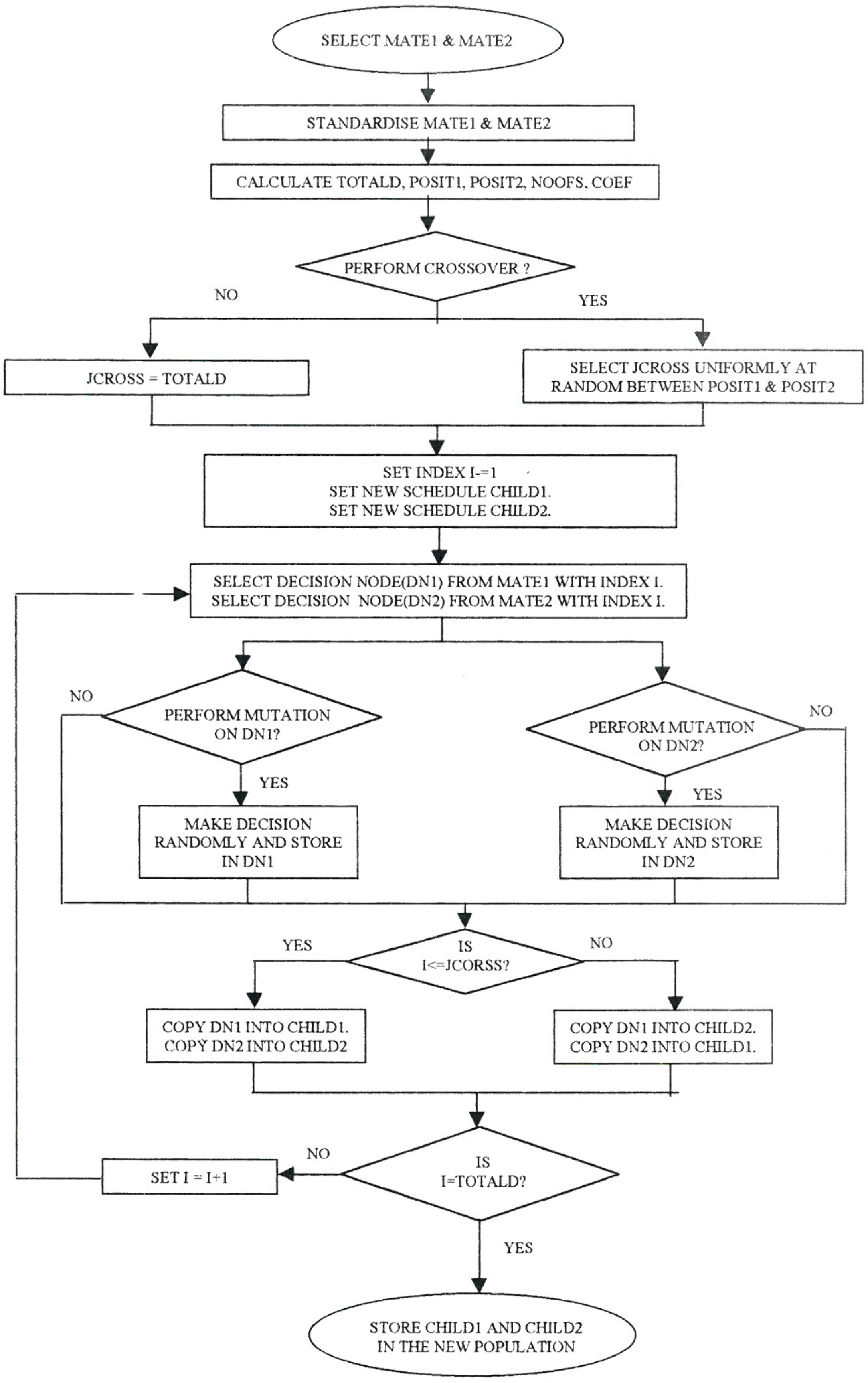

SELECT MATE1 & MATE2
STANDARDISE MATE1 & MATE2
CALCULATE TOTALD, POSIT1, POSIT2, NOOFS, COEF
PERFORM CROSSOVER ?
NO
YES
JCROSS = TOTALD
SELECT JCROSS UNIFORMLY AT RANDOM BETWEEN POSIT1 & POSIT2
SET INDEX I-=1
SET NEW SCHEDULE CHILD1.
SET NEW SCHEDULE CHILD2.
SELECT DECISION NODE(DN1) FROM MATE1 WITH INDEX I.
SELECT DECISION NODE(DN2) FROM MATE2 WITH INDEX I.
NO
PERFORM MUTATION ON DN1?
YES
MAKE DECISION RANDOMLY AND STORE IN DN1
PERFORM MUTATION ON DN2?
NO
YES
MAKE DECISION RANDOMLY AND STORE IN DN2
YES
IS I<=JCORSS?
NO
COPY DN1 INTO CHILD1.
COPY DN2 INTO CHILD2
COPY DN1 INTO CHILD2.
COPY DN2 INTO CHILD1.
NO
SET I = I+1
IS I=TOTALD?
YES
STORE CHILD1 AND CHILD2 IN THE NEW POPULATION

Figure 4. Flowchart for Crossover and Mutation

During simulation, the schedules can get modified, e.g. a decision may specify a product on a particular machine when, during actual simulation, the order is already completed. In such cases, a new random decision is made and stored in the schedule.

EVALUATION CRITERIA:

Each product has a due date (DD) by which its order has to be complete. The evaluation criteria can be:

TOTAL TIME (TT):

The total time(TT) is the time when all the product orders are complete. This time is obtained by simulating the schedule until all the orders are complete.

TOTAL DELAY TIME (TDT):

Each product has a due date. The time at which the product order gets completed is the order completion time (OCT).

$ODT_I = OCT_I - DD_I$

I= product number

ODT_I = order delay time of product I.

$TDT = \sum ODT_I$

MAXIMUM TARDINESS:

Another fitness criteria could be

100000-Max Tardiness

FITNESS SCALING

Fitness is defined as

FITNESS = 100000 – TDT

The lesser the total delay time , the higher is the fitness of the schedule. If left to normal selection rule, the extraordinary schedules would take over a significant proportion of the finite population in a single generation, leading to premature convergence. Late in a run, there may be significant diversity within the population, however the population average fitness may be close to the population best fitness. If this is left alone, average members and best members get nearly the same number of copies in future generations and survival of the fittest becomes a random walk among the mediocre.

Fitness scaling helps to overcome these obstacles. Linear scaling procedure is used.

NEW FITNESS = A* FITNESS + B

The coefficients A and B are determined by the following conditions. The average scaled fitness is set equal to the average raw fitness. The scaled maximum fitness is equal to FMULTIPLE*AVERAGE FITNESS, where FMULTIPLE is the number of expected copies desired for the best population schedule. FMULTIPLE is given a value of 2. The stretching required on average and maximum raw fitness values may cause the low fitness values go negative after scaling. In such cases, we do not scale to the desired FMULTIPLE. The raw and scaled fitness averages are kept equal and the minimum raw fitness is mapped to a scaled fitness of zero.

GA PARAMETERS

The parameters for the genetic algorithm are selected from research papers published in this field. Jain (1997) suggested a population size of 80 and FMULTIPLE equal to 1.9. Goldberg (1989) suggested a mutation probability inversely proportional to population size. The parameter values thus selected are

FMULTIPLE = 2

MAXPOP = 80

PCROSS = 0.6

PMUTATION = 0.0125

GENETIC ALGORITHM

The various steps in the genetic algorithm are:

1. INITIALIZATION

a. Set the values of the GA parameters. Set the values of MAXPOP, PMUTATION, PCROSS ,FMULTIPLE and MAXGEN, the number of generations to be created.

b. Read the database of the manufacturing system to be processed. Read the number of products, number of original machines, changeover times, setup times, production rates and due dates.

c. Create an initial population of schedules of size MAXPOP by simulation. This is called old pop.

d. Calculate the fitness values of all the schedules and scale the raw fitness values.

2. Set generation no GENNO = 1.

3. NEW POPULATION

a. Apply the reproduction operator on the old pop to select two schedules for mating.

b. Standardize the two mates into a common form.

c. Apply the surrogate crossover operator to obtain two new schedules by crossing the two mates.

d. Mutate each of the the two new schedules by applying the adaptive mutation operator.

e. Store the two new schedules hence obtained in the new population called newpop.

f. Repeat the above steps from a) to e) until the size of the new population is equal to MAXPOP.

g. Simulate all the schedules of the newpop to get their raw fitness values.

h. Scale the raw fitness values by using linear scaling.

i. Set oldpop equal to newpop.

4. If current GENNO < MAXGEN ,MAXGEN being the number of generations to be created, increment GENNO number by one and go to step 3. Else the fittest schedule of the current population is the best schedule .

3. Sample Schedule: GA Application

A sample schedule generated by GA application to a typical reconfigurable production system problem is presented in Figure 5. We illustrate the time dimension, machine status and quantities of remaining product orders. A discrete event modelling framework is used to conveniently represent the status of the system at the

```
TIME  M[1,1,1]  M[2,1,1]  P1     P2     P3     P4     P5     P6     P7
0     0         0         10771  9016   11297  19708  6840   18746  10882
        Decision :  Put Product 1 on M[1,1,1]
                    Put Product 5 on M[2,1,1]
TIME  M[1,1,1]  M[2,1,1]  P1     P2     P3     P4     P5     P6     P7
133   1         0         10771  9016   11297  19708  6840   18746  10882
        Completion : Setup of Product 1 on M[1,1,1]
TIME  M[1,1,1]  M[2,1,1]  P1     P2     P3     P4     P5     P6     P7
148   1         5         10576  9016   11297  19708  6840   18746  10882
        Completion: Setup of Product 5 on M[2,1,1]
TIME  M[1,1,1]  M[2,1,1]  P1     P2     P3     P4     P6     P7
636   1         0         4224   9016   11297  19708  18746  10882
        Completion:  Required Quantity of Product 5 made
        Decision:   Put Product 6 on M[2,1,1]
TIME  M[1,1,1]  M[2,1,1]  P1     P2     P3     P4     P6     P7
740   1         6         2872   9016   11297  19708  18746  10882
        Completion: Changeover from Product 5 to Product 6 on M[2,1,1] complete
TIME  M[1,1,1]  M[2,1,1]  P2     P3     P4     P6     P7
961   0         6         9016   11297  19708  15653  10882
        Completion: Required Quantity of Product 1 made
        Decision:   Break  M[1,1,1] into 2 parts
TIME  M[1,1,2]  M[1,2,2]  M[2,1,1]  P2     P3     P4     P6     P7
1041  0         0         6         9016   11297  19708  14535  10882
        Completion: Setup of M[1,1,2] complete
        Decision :  Put Product 7 on M[1,1,2]
                    Put Product 6 on M[1,2,2] after its setup is complete. This setup would be complete
                    at TIME=1121.
```

Note: Decisions regarding selection of products for sub-production lines are made when the setup of first sub-production line is complete.

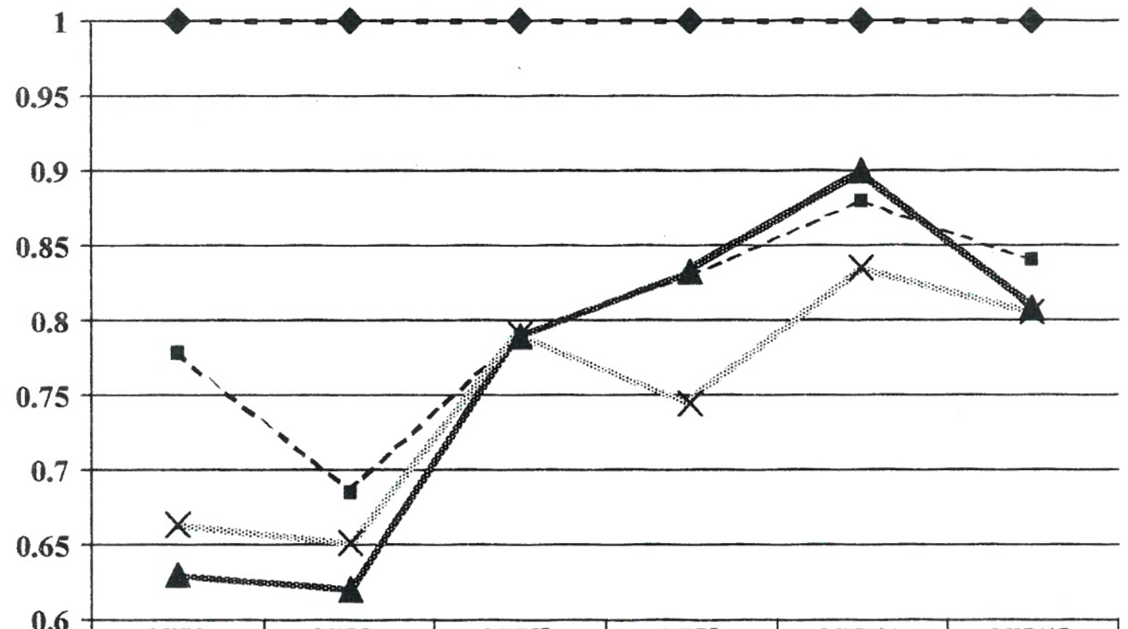
```
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P2      P3       P4       P6      P7
1111   7         0         6           9016    11297    19708    13555   10882
         Completion: Setup of Product 7 on M[1,1,2] complete
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P2      P3       P4       P6      P7
1183   7         6         6           9016    11297    19708    12543   10556
         Completion: Setup of Product 6 on M[1,2,2] complete. It had started at TIME = 1121.
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P2      P3       P4       P7
1861   7         0         0           9016    11297    19708    7505
         Completion: Required Quantity of Product 6 made.
         Decision:              Put Product 3 on M[1,2,2]
                                Put Product 2 on M[2,1,1]
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P2      P3       P4       P7
1933   7         3         0           9016    11297    19708    7183
         Completion: Changeover from Product 6 to Product 3 on M[1,2,2] complete.
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P2      P3       P4       P7
2003   7         3         2           9016    10821    19708    6866
         Completion: Changeover from Product 6 to Product 2 on M[2,1,1] complete.
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P3      P4       P7
2754   7         3         0           5749    19708    3485
         Completion: Required quantity of Product 2 made.
         Decision:   Put Product 4 on M[2,1,1]
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P3      P4       P7
2856   7         3         4           5061    19708    3026
         Completion: Changeover from Product 2 to Product 4 on M[2,1,1] complete
TIME   M[1,1,2]  M[1,2,2]  M[2,1,1]      P3      P4
3529   0         3         4           522     12983
         Completion: Required quantity of Product 7 made.
         Decision:   Combine M[1,1,2] and M[1,2,2] into M[2,1,1]
TIME   M[1,1,1]  M[2,1,1]      P3      P4
3595   0         4             522     12318
         Completion: Combination of M[1,1,2] and M[1,2,2] into M[1,1,1] complete
         Decision:   Put Product 3 on M[1,1,1]
TIME   M[1,1,1]  M[2,1,1]      P3      P4
3729   3         4             522     10978
         Completion: Setup of Product 3 on M[1,1,1] complete
TIME   M[1,1,1]  M[2,1,1]      P4
3764   0         4             10630
         Completion: Required quantity of Product 3 made.
         Decision:   Put Product 4 on M[1,1,1]
TIME   M[1,1,1]  M[2,1,1]      P4
3886   4         4             9410
         Completion: Changeover from Product 3 to Product 4 on M[1,1,1] complete.
TIME   M[1,1,1]  M[2,1,1]
4409   0         0
         Completion: Required quantity of Product 4 made.
```

Figure 5. Sample Schedule Obtained Through GA Application On A Typical Problem

beginning or at the end of each event associated with any decision. The beginning of the decision is represented by *Decision* and completion of any process activity is represented by the word *Completion.* As can be seen, some specific completions give rise to new decisions. For instance, the completion of a product order initiates the next decision regarding assignment of the remaining products on a reconfigured system.

Results and Inferences

Random data sets were created to model different production scenarios. The various data items such as production rates, due dates , setup times , demand, etc. were selected randomly from uniform distributions, each distribution being given a specific upper and lower limit. The operation of production system with these databases (production scenarios) was simulated using standard scheduling rules SPT (no breakups allowed) and parameters total tardiness and maximum tardiness were computed. The operation of production system with these databases was also simulated on three variants of the GA , viz., GA with no breakups allowed, GA with only phased breakups allowed and GA with only simultaneous breakups allowed. The number of generations for GA in each case was set at 10.

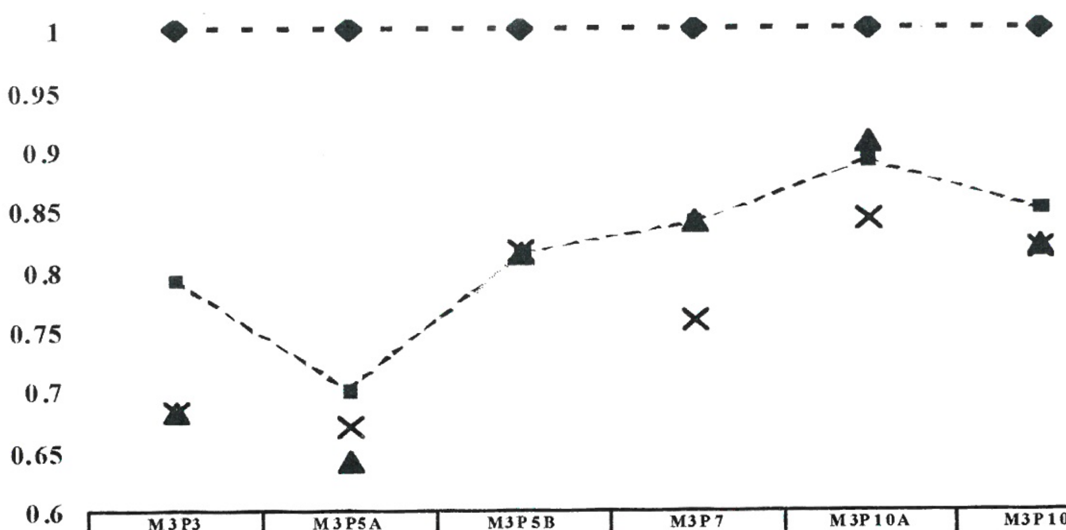


	M3P3	M3P5A	M3P5B	M3P7	M3P10A	M3P10B
◆ SPT	1	1	1	1	1	1
▲ GA1	0.63	0.62	0.789	0.832	0.9	0.808
■ GA2	0.779	0.685	0.79	0.83	0.88	0.84
✕ GA3	0.663	0.652	0.79	0.744	0.835	0.805

Figure 6. Graph of Ratio of Total Tardiness of A Scheduling Strategy to That of SPT for Alternative Re-configurable Systems

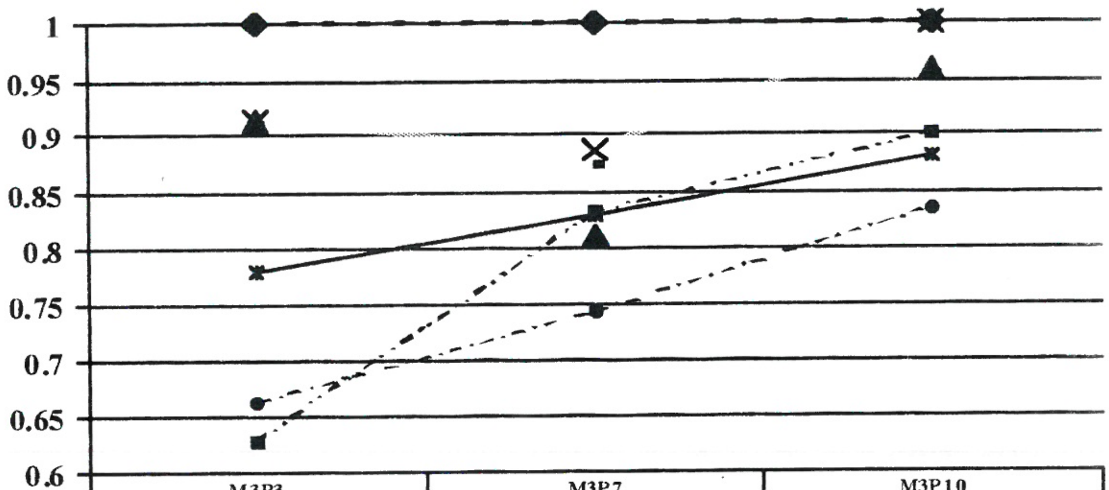
Total tardiness and maximum tardiness were computed. The results of. GAs were compared with SPT on the basis of total tardiness and maximum tardiness.

Further to this, the ratio of the number of production lines to the number of products was varied in different databases. Another factor which was studied was increasing the range of possible demands for products, the range of possible production rates and the range of possible due dates. A production system which has highly different demands and due dates for the various products and in which the production lines have different production rates for different products, was thus studied. The results of these studies are shown in Figure 6, Figure 7, Figure 8. In the Figures, the notation for Alternative Production System Configurations is MXPY where MX implies X number of original unbroken production lines and PY implies Y number of product types. Note that in the Figures, the **ratio** of total tardiness or maximum tardiness, as in the definition of the fitness function of the GAs , of schedules obtained by GAs to the corresponding tardiness of SPT schedule is plotted. This implies the lesser this ratio, the better the performance of the scheduling strategy compared to SPT is.



	M3P3	M3P5A	M3P5B	M3P7	M3P10A	M3P10B
◆ SPT	1	1	1	1	1	1
▲ GA1	0.683	0.64	0.813	0.84	0.906	0.82
■ GA2	0.792	0.7	0.815	0.839	0.89	0.85
✕ GA3	0.683	0.67	0.815	0.758	0.842	0.817

Figure 7. Graph of Ratio of Maximum Tardiness of A Scheduling Strategy to That of SPT for Alternative Re-configurable Systems



	M3P3	M3P7	M3P10
◆ SPT	1	1	1
▲ GA1A*	0.913	0.811	0.96
■ GA2A	0.912	0.875	0.955
✕ GA3A	0.9138	0.888	1
■ GA1B	0.63	0.832	0.902
✱ GA2B	0.779	0.83	0.88
● GA3B	0.663	0.744	0.835

Figure 8. Graph of Ratio of Total Tardiness of A Scheduling Strategy to That of SPT for Alternative Re-configurable Systems

Figure 6 illustrates the results of our experiments for alternative data sets (represented by M3P3, M3P5A, M3P5B, etc.). As we go from left to right, the number of products is increasing for the same number of machines (hence increasing the scheduling complexity and the number of alternative options available for allocation). Thus, each data set represents a specific production system configuration, standing for an increased level of product allocation complexity. For these alternative configurations, the graph indicates the performance of four alternative scheduling strategies - SPT , GA1, GA2, GA3. Under SPT, the scheduling follows the conventional shortest processing time rule on the available product and machines over the planning period. GA1 refers to the GA application (instead of SPT) where the production system cannot be re-configured as no breakups are allowed. GA2 refers an GA application where the production system can be re-configured at any event during the evolution of the system over a given planning period by phased breakups only (as explained earlier). GA3 represents an GA application where the system can be re-configured at any event by simultaneous breakups only. The measure of performance is based on total tardiness represented in the appropriate GA fitness function, to help visualize the relative performance of alternative scheduling strategies. As can be seen, the performance of the GA strategies is significantly superior to the conventional SPT rule. Also, it can be noted that as the system reconfiguration and product allocation options multiply, the relative performance of GA3 is mostly superior as compared to that of GA2. This implies that simultaneous re-configuration strategy is likely to be superior to phased re-configuration strategy. Figure 7 supplements our results for the case of fitness function involving maximum tardiness.

Figure 8 illustrates the effect of increasing variability in the production system with regard to production rates, demand and due dates for the different products. In GA1A, A stands for data sets created with one set of ranges of possible values of data items (production rates, demand, due date) while in GA1B, B refers to data sets created with larger range of possible values of data items. The performance of GA strategies appears to be sensitive to both the relative complexity and relative variability in the system. However, it is clear that GA based scheduling strategies consistently perform better than SPT. Within the GA strategies, the simultaneous re-configuration strategy mostly offers a good opportunity for tardiness reduction.

4.Conclusions

We have performed GA applications on a wide variety of alternative production system configurations offering various re-configuration options. A sample of these was shown here to highlight the key results. The GA application essentially offers an effective platform for dynamic re-configuration with integrated scheduling (DRIS). The results indicate that the schedules generated by GA are ever better in terms of total tardiness as well as of maximum tardiness as compared with the schedules obtained by conventional heuristics. Therefore it may be inferred that if the scheduling objective is to minimize either total tardiness or maximum tardiness, GA may offer better solutions. As the range of possible values of data items (production rates, demand and due dates) gets wider, GA with simultaneous breakups allowed, gives better schedules than GA with no breakups allowed. It may be inferred from this observation that the dynamic re-configuration of production lines as proposed in this paper may provide additional advantages while dealing with larger ranges of possibilities. As the range of possible values of production rates, demand and due dates gets larger and the ratio of number of products to number of machines is also higher, GA with simultaneous reconfigurations allowed, offered better schedules than the rest. This indicates that DRIS concepts may be implemented with various re-configuration option levels and GA may be used for integrated scheduling, to exploit the underlying flexibility.

The DRIS concepts demonstrated here can offer various opportunities for dynamic resource management leading to Agile Enterprises. In our opinion DRIS concepts are likely also to be quite useful in the domain of dynamic supply chains. The GA application on the DRIS model also has relevance to the service industry, for instance the effective management of teams of multi-skilled workers.

REFERENCES

1. AXELROD, R., **An Evolution of Strategies in the Iterated Prisoner's Dilemma**, in L. Davis (Ed.) Genetic Algorithms and Simulated Annealing, MORGAN KAUFMANN PUBLISHERS, 1987.
2. BOOKER, L, GOLDBERG, D. and HOLLAND, J., **Classifier Systems and Genetic Algorithms**, ARTIFICIAL INTELLIGENCE, 40, 1989, pp. 235-282.

3. BALAKRISHNAN, J. and JOG, P., **Manufacturing Cell Formation Using Similarity Coefficients and A Parallel Genetic TSP Algorithm: Formulation and Comparison**, MATHEMATICAL COMPUTER MODELLING, 21, 1995, pp. 61-73.
4. CHENG, R., GEN, M. and TOZOWA, T., **Minimax Earliness/Tardiness Scheduling in Identical Parallel Machine System Using Genetic Algorithms**, COMPUTERS AND INDUSTRIAL ENGINEERING, 29, 1995, pp. 513-517.
5. DAVIS, L., **Job Shop Scheduling with Genetic Algorithms**, Proceedings of the International Conference on Genetic Algorithms and Their Applications, Carnegie Mellon University, Pittsburgh, PA, 1985, pp.136-140.
6. GOLDBERG, D., **Genetic Algorithms in Search, Optimization and Machine Learning**, ADDISON- WESLEY, 1989.
7. GUPTA, M.C., GUPTA, Y. P and KUMAR, A., **Minimizing Flow Time Variance in A Single Machine System Using Genetic Algorithms**, EUROPEAN JOURNAL OF OPERATIONS RESEARCH, 70, 1993, pp. 289-303.
8. HOLLAND, J., **Adaptation in Natural and Artificial Systems,** UNIVERSITY OF MICHIGAN PRESS, 1975.
9. JAIN, A. K. and ELMARAGHY, H. A., **Production Scheduling/Rescheduling in Flexible Manufacturing**, INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH, 35 (1), 1997, pp. 281-309.
10. JOG, P., SUH, J. and VAN GUCHT, D., **Parallel Genetic Algorithms Applied to the Travelling Salesman Problem**, SIAM JOURNAL OF OPTIMIZATION, 4, 1991, pp. 515-529.
11. LIEPINS, G. E., HILLIARDS, M.R., PALMER, M. and MORROW, M., **Greedy Genetics: Genetic Algorithms and Their Applications**, Proceedings of the Second International Conference on Genetic Algorithms, Cambridge, MA, 1987.
12. LEE, C.Y. and CHOI, J.Y., **A Genetic Algorithm for Job Sequencing Problems With Distinct Due Dates and General Early-Tardy Penalties**, COMPUTERS AND OPERATIONS RESEARCH, 22, 1995, pp. 857-869.
13. DOUMEINGTS, G., VALLESPIER, B. and CHEN, D., **Methodologies for Designing CIM Systems: A Survey**, COMPUTERS IN INDUSTRY, Vol. 25, 1995, pp. 255 - 262.
14. LEE, C.Y. and KIM, S. J., **Parallel Genetic Algorithms for the Earliness-Tardiness Job Scheduling Problem With General Penalty Rates**, COMPUTERS AND INDUSTRIAL ENGINEERING, 28, 1995, pp. 231-243.
15. WADHWA, S. and BROWNE, J., **Modelling FMS With Decision Petri Nets**, INTERNATIONAL JOURNAL OF FLEXIBLE MANUFACTURING SYSTEMS, Kluwer Academic Publishers, 1, 1989, pp. 255-280.
16. WADHWA, S. and BHAGWAT, R., **Judicious Increase in Flexibility and Decision Automation in Semi-computerized Flexible Manufacturing (SCFM) Systems**, STUDIES IN INFORMATICS AND CONTROL, Vol. 7, No.4, 1998, pp. 329-342.