JOB SHOP SCHEDULING USING ARTIFICIAL IMMUNE SYSTEM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR DEGREE IN BACHELOR OF TECHNOLOGY IN MECHANICAL ENGINEERING

BY

DEEPAK KUMAR MAHAPATRA

ROLL NO-108ME004

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA

2012

JOB SHOP SCHEDULING USING ARTIFICIAL IMMUNE SYSTEM

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR DEGREE IN BACHELOR OF TECHNOLOGY IN MECHANICAL ENGINEERING

BY

DEEPAK KUMAR MAHAPATRA

ROLL NO-108ME004

GUIDED BY: PROF. S.S. MAHAPATRA

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA

2012

NATIONAL INSTITUTE OF TECHNOLOGY

CERTIFICATE

This is to certify that the thesis entitled "Job Shop Scheduling Using Artificial Immune System" submitted by Deepak Kumar Mahapatra in partial fulfillment of the requirements for the award of Bachelor in Technology Degree in Mechanical Engineering at National Institute of Technology, Rourkela (Deemed University), is an authentic work carried out by him under my supervision. To the best of my knowledge, the matter embodied in the thesis has not been submitted to any University/ Institute for the award of any Degree or Diploma.

Date:

Prof. S.S. Mahapatra

Department of Mechanical Engineering
National Institute of Technology,
Rourkela- 769008

ACKNOWLEDGEMENT

I avail this opportunity to extent my hearty indebtedness to my guide Prof. S .S. Mahapatra, Mechanical Engineering Department, for his valuable guidance, constant encouragement and kind help at different stages for the execution of this dissertation work.

I also express my sincere gratitude to Prof. K. P. Maity, Head of the Department, Mechanical Engineering, for providing valuable departmental facilities and Prof. S. K. Sahoo and Prof. C. K. Biswas, for constantly evaluating me and for providing useful suggestions.

Submitted By:

Deepak Kumar Mahapatra

Roll no- 108ME004

Department of Mechanical Engineering

National Institute of technology,

Rourkela - 769008

CONTENTS

ABSTRACT

Efficiency in job shop scheduling plays an important role when a large number of jobs and machines are considered. The job shop scheduling problems are one of the NP hard problems. Many heuristic methods give solutions with near optimal results. This work deals with the job shop scheduling using Artificial Immune System. Operation based representation is used to decode the schedule in the algorithm. The mutations used in the algorithm are inverse mutation and pair wise exchange mutation and a receptor editing process is also used. A C++ code was generated to use the algorithm for finding the optimal solution. The input parameters are operation time and operation sequence for each job in the machines provided. This work used the makespan values of the schedules to compare the results.

INTRODUCTION

Job shop scheduling is the sequencing of the different operations of a set of jobs in the different time intervals of a set of machines. These types of problems are found in flexible manufacturing systems, production planning, computer design, logistics etc. Scheduling can be categorized into three types:1. Open shop scheduling, 2.Job shop scheduling, 3. Flow shop scheduling.

1. Open shop scheduling: In this type of scheduling problems there are a set of m jobs and n machines as in other types but in this case there is no constraint in the order of the operation of each job. This can be solved in polynomial time for 2 machines but for more than 2 machines this problem is NP hard.

2. Flow shop scheduling. In flow shop scheduling problems there is also a set of m number of jobs and n number of machines but here the schedule has to follow a particular sequence of operations for each job. A minimum idle time and a minimum of waiting time are the constraints in the continuous flow of processes.Flow shop scheduling problems are generally found in production facilities. The flow shop scheduling problem is ageneralized version of the job shop scheduling problem for flexible manufacturing systems. Here each machine has the ability to perform more than one operation for a particular job. In classical JSP each operation is processed on a predefined machine but each operation in the FJSP can be processed any of the available machines. Thus the scheduling or routing of jobs and operations in flowshop scheduling problems is a very difficult task. Moreover, it is a NP hard problem whose solution can’t be found in polynomial time and is a more general form of job shop scheduling problem.

3. JOB SHOP SCHEDULING:

In a job shop scheduling problem it consists of a set of jobs (let's say m numbers) and set of machines (let's say n numbers). Each job has a number of operations and particular sequence of machines which it has to follow. Here each operation can only be done in one machine i.e. each machine can perform a single operation at a time.

For these types of problems no efficient solution algorithm is found yet but some of the algorithms developed give the best results so far in most problems. These types of problems are one of the well known hardest combinatorial optimization problems. In the past three decades many researchers have worked hard on these types of problems and a lot of solutions are derived. Some of these types of methods are: dispatching rules, tabu search, simulated annealing and genetic algorithm etc. As the computing technology has advanced a lot, so it is important in these days to optimize the schedules in a better way with some extra computation. The artificial immune system algorithm approach is one such kind of attempt. The objective is to obtain better schedules using artificial immune system than traditional heuristics methods.

The use of artificial immune system in a job shop scheduling problem is very difficult. The most difficult part is to encode the solution into an antibody and to represent it in a form which can be easily operated for further operations. It can't be simply represented in a binary form, which may lead to infeasible solutions.

Our objective is to generate such type of a schedule in job shop scheduling process which will minimize the time in which all the operations of every job will be completed i.e. the length of the schedule.

The Scheduling of the job shop provides a set of resources to tasks over time. In the past years vast amount of research have been done on operational research. This mainly focuses on finding ways of giving jobs to machines such that it meets certain criteria and an objective function is optimized. Till now a 100 percent perfect method has not been found to get the optimal solution in all types of job shop scheduling.

The constraints for a job shop scheduling problems are:

- A job must not visit the same machine more than once.
- There are no precedence constraints on operations of different jobs.
- Operations can't be interrupted.
- Each machine can process only one job at a time.
- Release time and due dates are not specified.
- Each job should go through a particular sequence of operations as predefined.

Objective function:

Each optimization problem must have an objective function which has to be either minimized or maximized in order to get a solution. In this case the objective function is the make span value or the length of the schedule. The make span value can be defined as follows: Here each job has a no operations and each operation has a particular make time. When these make times are arranged in the sequence of the schedule then each machine gets one particular make time. Out of all the machines the machines which have the maximum make time that is the make span value.

The Job shop scheduling problem formulation with makespan as the objective is as follows:

Min $\max_{1\le k\le m}\{\max_{1\le k\le n}\{C_{ik}\}\}$

s.t. $C_{ik} - t_{ik} + Z(1-a_{ihk}) >= C_{ik}$, **i=1.2….,n and h,k=1,2,…,m**

$C_{jk} - C_{ik} + Z(1-x_{ijk}) >= t_{jk}$, **i,j =1.2….,n and k=1,2,…,m**

$C_{ik}>=0$,**i=1,2,…,n and k=1,2,….,m**

X_{ijk}**=0 or 1, I,j=1,2,…,n and k=1,2,…,m**

C_{jk}**= Completion time of job j on machine k**

T_{jk}**= the processing time of job j on machine k**

Z is a very large positive number.

$$a_{ihk}=\begin{cases}1, if\ operation\ of\ job\ i\ on\ machine\ h\ preceeds\ that\ on\ machine\ k\\ 0, others\end{cases}$$

$$x_{ijk}=\begin{cases}1, if\ \ job\ i\ preceeds\ job\ j\ on\ machine\ k\\ 0, others\end{cases}$$

LITERATURE REVIEW

Many heuristic approaches are developed in the past decades by researchers for solving job shop scheduling problems. Some of these methods are dispatch rules, Tabu search (TS), simulated annealing, particle swarm optimization, genetic algorithm and artificial immune system.

Job shop scheduling problems are NP hard. Bruker [1] and Garey [2] stated this and thus getting solution of these types of problems are very difficult.Brandimarte (1993) was the first to apply this heuristic method to solve job shop scheduling problems[3]. Carlier and Pison [4] and Bruker [5] found the solution of small size problems using branch and bound methods. For solving large sized problems Blazewicz[6] developed the method of efficient local search. The results (i.e. the minimum makespan) in his method were found at least for one of the preferred schedules and thus reducing the search efforts.

Erscher et al. a branch and bound method with three parts. Step 1 is to calculate the lower bound, step 2 is branching and step 3 is node elimination[7]. Hurik, Jarisch and Thole (1999) and Dazere-Peres used the different tabu search methods for job shop scheduling problems.[8]

Mastrolilli and Gambardella (2000) worked on the neighborhood functions for Flexible job shop scheduling which can be used in Meta heuristic optimization techniques. This method got better results than any other methods in computational results and solution quality.[9]

Tavakkoli-Moghaddam et al. presented a hybrid method for stochastic job shop scheduling minimizing the difference between the delivery and the completion times of jobs as well as related operational or idle cost of machines. Simulated annealing method was used in their work. Initial feasible solutions were generated by neural network and the performance quality of the initial solution was enhanced using simulated annealing method. [10]

Daeyoungchung,et al. stated a heuristic algorithm that addresses the problem by solving series of subproblems to optimality. Two steps were used in their algorithm and those are step 1: for improving the sequence of operations and step 2 for picking out the operations to be sub contracted on bottleneck machines.[11]

D l Santos et al. presented global lower bounds for FSMP makespan problems which may be used to assess the quality of heuristic solutions when the optimal solution is known.[12]

JinweiGu, et al. proposed a novel parallel quantum genetic algorithm for stochastic job shop scheduling. Their objective was to minimize the expected value of makespan where the processing times are subjected to be independent normal distribution[13]. QuinNiu, Bin Jiao, Xingsheng Guused the particle sworm optimization combine with genetic operators to solve the Job shop scheduling problems with fuzzy processing time .explain fuzzy processing time.[14]

J timmis et al. worked on the details of the three types of AIS algorithms i.e. the clonal selection, immune network and negative selection algorithms[15]. JieGao, Linang Sun, Mitsuo Gen woked on a new hybrid genetic and variable neighborhood search algorithm. To strengthen the search ability the individuals of GA are first improved by a variable neighbourhood descent.[16]

Q Zhang, et al. worked on a genetic algorithm with tabu search procedure for job shop scheduling problems with transportation constraints and bounded processing times. The FJSSP with transportation constraints involved by critical handling resources and the processing duration of jobs are limited by lower and upperbounds.[17]. EbruDemirkol, Sanjay Mehta and RehaOzsoy stated randomly generated test problem set for minimization of makespan (Cmax) problems in flow shop and job shop.[18]

METHODOOGY ADOPTED

METHODOLOGY ADOPTED:

The methodology to be used to solve this problem is the artificial Immune system. This system uses a set of Libraries i.e. a set of possible schedules to create an antibody a possible complete schedule. An antigen is a randomly generated complete schedule which will be used further to compare with the antibodies. If the antibody gives better solution, then the antigen should be updated with the antibody. Then for a particular no of iterations as defined by the user decode the antibody and local search to improve it. Then generate N clones of the antibody. Mutate among the antibodies created randomly to get better segments of the schedule. Update the library with better segments. Continue this operation till all the iterations are done.
The final antigen obtained is our optimal solution.

The types of representation which are used to encode the schedule are:

- Operation based representation
- Job based representation
- Preference list based representation
- Job pair relation based representation
- Priority rule based representation
- Disjunctive graph based representation
- Completion time based representation
- Machine based representation
- Random keys representation

Let us take a simple 3 job 3 machine problem to describe the all above representations: We have 3 jobs whose operation sequence and operation times are as given below:

JOBS	OPERATIONS		
	1	2	3
1	2	3	2
2	1	4	2
3	2	3	1
JOBS	MACHINE SEQUENCE		
1	1	2	3
2	3	2	1
3	2	3	1

Table 1: A sample job shop problem

Operation based representation:

As the name suggest this representation uses a string of numbers or integers to represent the sequence of operations of each job. Because of the existence of precedence constraints all the possible permutations doesn't represent a feasible schedule. The number of integers in the antibody of an m job and n machine job shop problem is m*n.

To decode the schedule from the generated antibody we have to follow the steps below:

1. The integers in the antibody represent the job numbers.
2. After the job sequences are generated the operation sequence is decoded from the operation sequence list. The first operation in the list is scheduled first, then the second and so on.
3. Each operation is allocated to the best available processing time for the corresponding machine which the operation is allocated.

The schedule generated by the above process guarantees an active schedule. In the example we have considered a random antibody generated in this type of representation will be as [111233232]. Here each number is decoded as O_{jim} where j represents the job number, i gives the operation number and m the machine. So here the first 1 is the 1 st operation of job 1 in machine 1, the second 1 represents the 2^{nd} operation of job 1 in machine 2 and so on. Each number represents a unique operation.

Job based representation:

In this type of representation the antibody i.e. the schedule is generated in terms of the jobs. Here any combination of the jobs gives a schedule. The schedule is decoded in the following way. The first gene in the antibody gives the job which is to be processed first. Then all the operations of this job are allocated to the best available time of the machines according to the operation sequence as specified in the problem. Similarly all the jobs are processed according to their preference. So any permutation of the jobs gives a feasible schedule.

In the problem given in table 1 the antibody generated by this type of representation will be [321]. So first we have to complete all the operations of job 3 then of job 2 and after that job 1. And their operation sequence are [231] for job 3 [321] for job 2 and [123] for job 1.

In a m job n machine problem the length of the antibody in this type of representation will be m. The antibody will be $[m_1,m_2,\ldots,m_m]$. [19]

Preference list based representation:

This representation was proposed by Davis for a kind of scheduling problem[12]. In a m job n machine problem the antibody consists of n sub antibodies of length m for each machine. The operations are processed in the relevant machines. The sub antibodies do not give the operation sequence they are the preference list. The schedule is generated through simulation from the preference list and the operation sequence list.

Suppose a generated antibody is [(123) (123) (231)]. The genes are the preference list for the machines 1, 2 and 3 respectively. The first preference are for the jobs 1 on machine 1, job 1 on machine 2 and job 2 on machine 3. So first job 1 on machine 1 is schedule and there after job 1 on machine 2 and job 2 on machine 3. Now again the preference list is generated. Now the preference are job 2 on machine 1,job 2 on machine 2 and job 3 on machine 3. So now job 2 is scheduled on machine 2 and job 2 on machine 1.now again according to preference list job 3 is processed on machine 2, 3 and 1 respectively from the preference list. At last job 1 is processed on machine 3. The Gantt chart for each step is shown in the figure.

The advantage of this type of representation is that it always gives a non delay schedule but a major disadvantage is that non delay schedules are always not optimal.

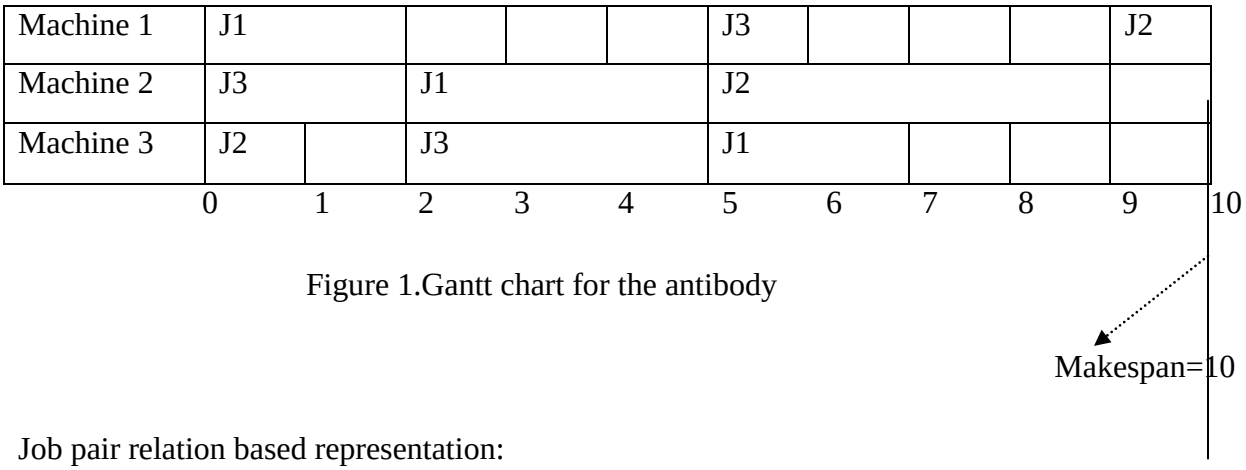


Figure 1.Gantt chart for the antibody

Job pair relation based representation:

Operation precedence:

JOBS	MACHINE SEQUENCE		
J1	M1	M2	M3
J2	M3	M2	M1
J3	M2	M3	M1
Machine	JOB SEQUENCE		
M1	J2	J3	J1
M2	J3	J2	J1
M3	J1	J3	J2

Table 2: A feasible schedule using Job pair based representation.

In problem of a 3 job 3 machine the operation precedence constraints and a feasible solution is given in the table.

X_{ijm} is a binary variable which is used to decide the job precedence relation in a machine m.

$$X_{ijm}=\begin{cases}1, \text{if job i is processed before job j on machine m}\\ 0, in\ other\ cases\end{cases}$$

Now we take each job pair and find the Xijm values. Thus we can generate the schedule. Let for the job pair (j_1,j_2) the value of (X_{121},$x_{12}2$,x_{123}) are (001) and for job pair (j_2,j_3) the value of (X_{231},x_{232},x_{233}) = (100) and for job pair (j_1,j_3) the value of (X_{131},x_{132},x_{133})=(001). Now the schedule generated is show in the table 2.

The binary matrix representation is given as

$$\begin{matrix}(j_1, & j_2)\\ (j_2 & j_3)\\ (j_1 & j_3)\end{matrix}\begin{pmatrix}\mathrm{m}_1 & \mathrm{m}_2 & \mathrm{m}_3\\ \mathrm{m}_1 & \mathrm{m}_2 & \mathrm{m}_3\\ \mathrm{m}_1 & \mathrm{m}_2 & \mathrm{m}_3\end{pmatrix}:\begin{pmatrix}x_{121} & x_{122} & x_{123}\\ x_{231} & x_{232} & x_{233}\\ x_{131} & x_{132} & x_{133}\end{pmatrix}=\begin{pmatrix}0 & 0 & 1\\ 1 & 0 & 0\\ 0 & 0 & 1\end{pmatrix}$$

This is the most complex type of representation among the various types of representations in job shop scheduling.

Disjunctive Graph based representation:

A disjunctive graph is the graphical representation of a minimization problem in job shop scheduling. There are m*n+2 nodes in each graph where m is the number of jobs and n is the number of machines. Each node represents a unique operation. The first node represents the source and the last node is the sink. When the nodes are connected by solid arcs it is the conjunctive arcs and when these are connected by dotted lines these are called disjunctive arcs.

The graph connected with the solid lines gives the sequence in which the schedule will work. All the operations which are to be done in the same machine are connected by disjunctive arcs. The length of the arcs gives the operation time of each node. The arcs emanating from the source are all conjunctive and all have length as zero.

A feasible solution is represented by the selection of disjunctive arc from each pair such that the graph is acyclic. This type of selection determines the sequence of operations to be performed on that machine.

Let's take an example of an infeasible solution. Let (h, p) and (i, p) denote two consecutive operations that belong to job p and let (i, q) and (h, q) denote two consecutive operations that belong to job q. If under a given schedule operation (i, p) precedes operation (i, q) on machine i and operation (h, q) precedes operation (h, p) on machine h, then the graph contains a cycle with four arcs, two conjunctive arcs and two disjunctive arcs from different cliques. Such a schedule is physically impossible.

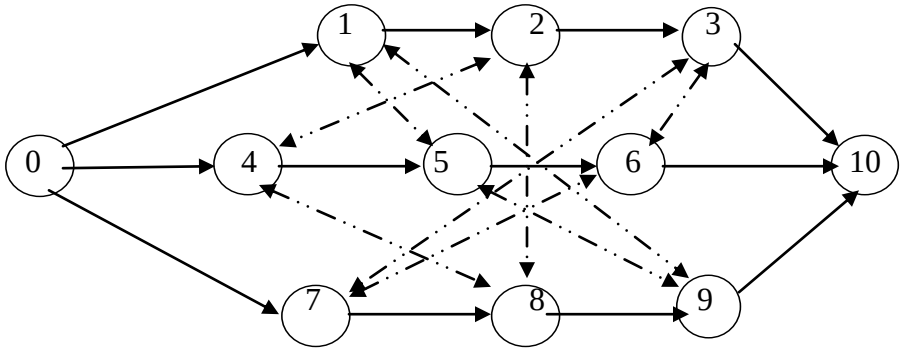


Figure 2. DISJUNCTIVE GRAPH

This type of representation uses a string of binary numbers to decode the schedule.

$$e_{ij}=\begin{cases}1, settle\ the\ orientation\ of\ disjunctive\ arc\ from\ node\ j\ to\ i\\0, settle\ the\ orientation\ of\ disjunctive\ arc\ from\ node\ i\ to\ j\end{cases}$$

An example of the antibody is

$e_{15}\ e_{19}\ e_{59}\ e_{24}\ e_{28}\ e_{48}\ e_{36}\ e_{37}\ e_{67}$

The generated antibody is [001100011]

Priority rule based representation: It is one of the frequently used method for job shop representation in various heuristics. For an m*n problem a string of n*m is generated and each entry represents one rule of set of pre specified priority dispatching rules.

Completion time based representation: In this type of representation the chromosome is ordered as a list of completion times of operations. This type is not used in many cases because it may result an illegal solution.

Machine based representation: In machine based representation the antibody is encoded as a sequence of machines and the sequence is generated with a shifting bottleneck heuristic based on the sequence.

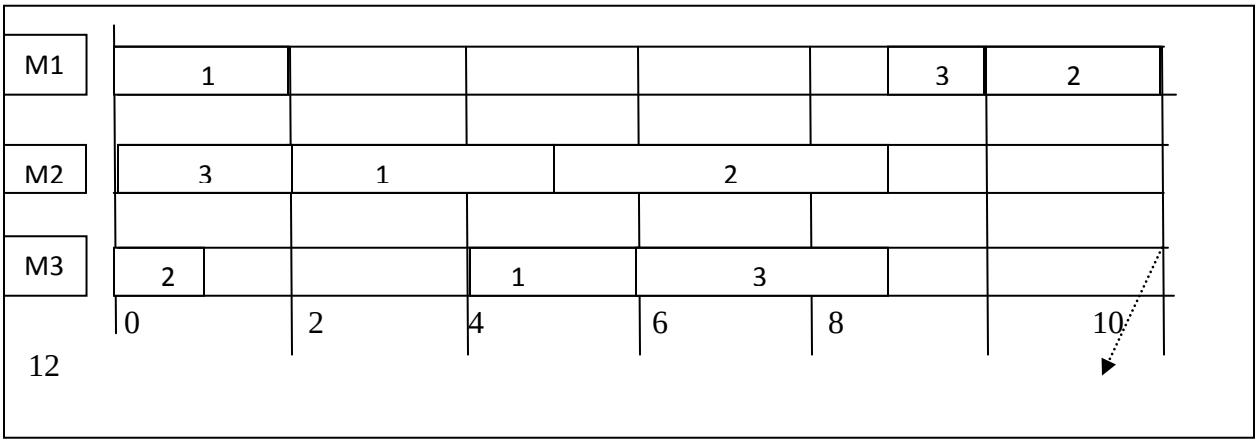
Random key representation: It uses a random number technique in which each gene has an integer part and a fraction part. The integer part gives the machine assignment for the job and the fraction parts represent the sequence of jobs in the machine.

Gantt chart:

Gant chart is the type of representation in bar chart used to show a feasible schedule in job shop scheduling problems. This was developed by Henry Gantt. This also gives the details about the precedence of the operations and also the sequence of operations of either jobs or machines.

Gantt chart is suitable for displaying the resulting schedule in a small problem but in a problem with large number of activities it is very difficult to represent the schedule. Gantt chart does not represent the relative size of the work elements or the total size of the project. So it becomes very difficult in some cases to compare between two projects with same number of completion time.

In a job shop scheduling problem the Gantt chart uses a bar chart to decode the schedule. Either of the job or the machines are represented in the y axis and in x axis we select the constraint which we used to represent our solution i.e. the makespan value or the completion time. Each activity here represents the operation in the particular machine as the row. The machine with highest makespan value gives the makespan of the whole schedule. If an antibody is given as [111233232] for the problem stated in table 1 then its Gantt chart will be as drawn below. It's a 3 job 3 machine problem.



Makespan value=12

Figure 3. GANTT CHART

From the Gantt chart drawn for the antibody the makespan value can be calculated by finding the maximum makespan among all the machines. Here M1 has a makespan value of 12, M2 and M3 each have a makespan value of 9. So the resulting makespan value of the makespan for the schedule is given as 12.

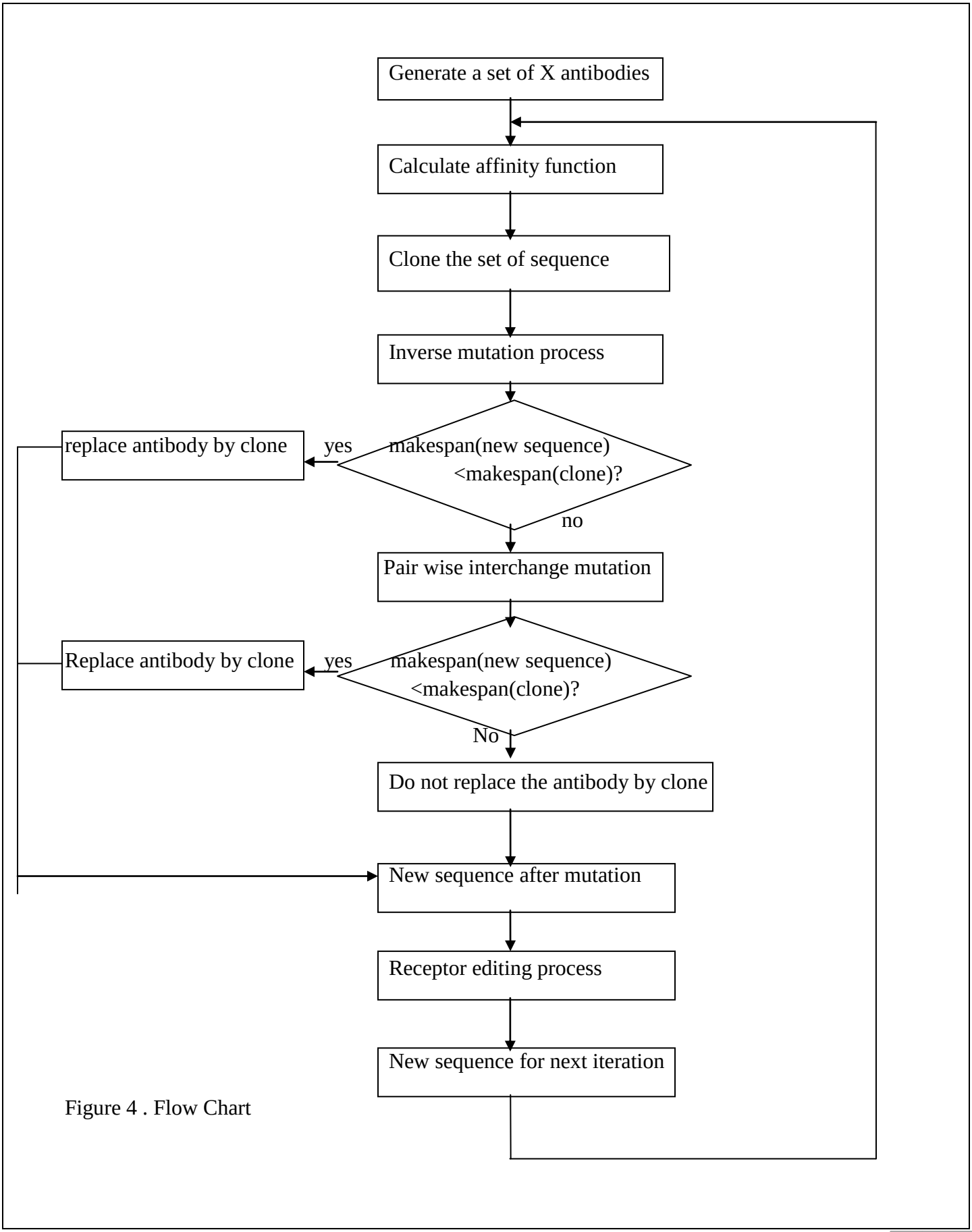


Figure 4 . Flow Chart

Artificial immune system:

Artificial immune system is the computational form derived from the biological immune system. In the biological immune system the foreign pathogens are blocked in the human body by the antibodies present in the body. The adaptive immune system helps to shield the body by recognizing the specific foreign cells protected before and letting the particular response to those foreign cells stay in the body for a little longer. Many researchers are working on the algorithm used for artificial immune system. So it's still an evolving field.

Artificial immune system for job shop scheduling:

In artificial immune system the antigen represents the best solution. First we generate a set of random antibodies. In our problem the antibodies represent a schedule of the job shop scheduling problem. After generating the initial population of antibodies we find the makespan of each of the antibodies and thus the affinity values. We find the affinity index value of the population which is used to calculate the number of clones to be generated. Then we create a clone population in which the antibody with best affinity index value is cloned to maximum numbers and the antibody with lesser value of affinity index is cloned to minimum numbers. Then the population is sorted according to the best makespan values. After generating the cloned population inverse mutation and pair wise exchange mutation are done. If the make span value after mutation is better than that is updated in the population. After the completion of mutation receptor editing process takes place. The receptor editing probability is initially given by the user. Then again taking the best values from the population the process continues. After several iteration the best antibody is generated, which is our final solution.

Initial population:

The initial population is defined as a set of randomly generated antibodies. These all antibodies should represent a feasible schedule. Using these antibodies we create the clone population by cloning each of the antibodies in the initial population to some proportion. Again this population is sorted according to their makespan values with the minimum in the first.

Receptor editing:

The editing in the antibodies of the cloned population after the mutation process is known as receptor editing. In this process we eliminate a number of worst affinity value antibodies from the population and add randomly generated antibodies in those places. The receptor editing probability multiplied by the clone population gives the number of worst valued antibodies which are to be replaced.

Mutation: For permutation representation of the antibodies several mutation operators are derived in the past decade. The different types of mutations are

- Inversion mutation
- Insertion mutation
- Displacement mutation
- Pair wise exchange mutation
- Shift mutation

1. Inverse mutation:

In this type of mutation process the antibody is inverted in between two positions. The two positions are randomly generated. If the upper bound random number becomes equal to the lower bound random number then we generate another random number till upper bound is higher than the lower bound. The maximum value of the upper bound in a m job n machine problem is (m*n) and minimum value of the upper bound will be 2 when lower bound is 1. Similarly the maximum value of the lower bound is (m*n-1) when upper bound is m*n and minimum value is 1.

Let's take an example of an antibody as [121322313] in a 3job 3 machine problem as given in the table 1. And suppose the randomly generated upper and lower bounds are 7 and 3. So now the string between the 3^{rd} position and 7^{th} position are to be inverted. So the antibody generated after the inverse mutation process is [123223113].

2. Pair wise exchange mutation:

In this type of mutation we randomly select two positions a and b between 1 and m*n in a m job n machine problem. Now the operations in positions a and b are interchanged. The new antibody string generated after mutation process is also a feasible solution.

In the antibody is [121322313] and the values of a and b are taken as 3 and 5 then the mutated antibody will be [123322113].

Insertion mutation:

Insertion mutation selects a gene at a random position in the antibody and then inserts it into another randomly generated position in the antibody.

If our antibody is [121322313] and the gene selected is 3^{rd} position i.e. 1 then insert it into 5^{th} position then new mutated antibody will be [123212313].

Displacement mutation:

In displacement mutation process it selects a substring between two random positions. Then it inserts the substring at a random position in the rest of the antibody.

In the antibody [121322313] if the substring is generated as [322] between 4^{th} and 6^{th} position and the insertion position is generated as 2 then our new string will be [132221313]. Insertion mutation can be considered as a special case of displacement mutation where the length of the substring is taken as 1.

Shift mutation:

Shift mutation process involves the shifting the gene of a particular position to a random position of right or left from the gene's position.

If the antibody is [121322313] and the selected gene is 5^{th} position i.e. 2 and we shift it to 2 positions in left then the new generated antibody will be [122132313].

Affinity function: The affinity an antibody or schedule is derived from the value of the makespan. The affinity of each antibody is given by:

$$affinity = \frac{1}{makespan}$$

The antibody with a higher makespan value has less affinity and the antibody with lower makespan value has greater affinity. The antibodies with greater affinity value will have more number of clones than the antibodies with lower affinity value as the number of clones is directly proportional to the affinity value.

Affinity index: The affinity index is used to calculate the exact number clone that has to be generated of an antibody. The number of clones is given by the affinity index multiplied by the clone population size. For calculating the affinity index values use the following method:

1. Calculate the makespan value thus the affinity value of each antibody in the initial population.
2. Calculate the summation of all the affinity values.
3. The affinity index is given by the ration of the individual makespan to the summation of the affinity values of the population.

The type of representation used in the generated C program is operation based representation. A code was generated in dev C++ to use the algorithm for solving job shop scheduling problems. The initial population size taken is 20.

Clone population size is 100.

Receptor editing probability = 20%

Iterations are continued till the optimal solution is found.

ALGORITHM FOR AIS:

1. Create a population of Z antibodies (Z is the parameter for antibody population size)
2. For each iteration use the following steps
3. Calculate the makespan of the antibodies in the population
4. Calculate the affinity value and thus the affinity index of each antibodies.
5. Clone each antibody in proportion of their affinity index value.
6. In the clone population apply inverse mutation to all the clones.
7. Calculate the makespan of the inverse mutated clone.
8. If the makespan of the inverse mutated clone is less than the original clone then replace the clone with the mutated clone, otherwise the clone remains the same.
9. Now apply pair wise exchange mutation to the clones.
10. Calculate the makespan values of the pair wise exchange mutated antibodies and of the original clone.
11. If the makespan value of the pair wise exchange mutated antibody is less than the makespan value of the clone then update the clone with the mutated antibody otherwise the clone remains the same.
12. Eliminate the worst antibodies as per the receptor editing probability value.
13. Create new random antibodies at the same number in place of the removed antibodies.
14. Now sort the clone population according to their affinity index values with the lowest makespan value first.

RESULTS AND DISCUSSIONS

A C++ code was generated in Dev C++ using the artificial immune system algorithm. The program was tested for 26 benchmark problems of various sizes as discussed below.The benchmark problems Ft06 and 10, LA01-15 due to Lawrence [19]and ORB01-10 due to Applegate and cook [20] weresolved and compared with the best known lower bound values for scheduling problems.The lower bound can be calculated by

LB=$\max_i\{\sum_{j=1}^{m} P(i,j)\}$ where P (i, j) be the processing time of job i (1 <i< n) at stage j,

The % Relative error is calculated in each of the problem if values are obtained above the lowerbound values. The error is given by RE%=100*(Makespan-LB)/LB, where LB is the best known lower bound. [21]In table 3 M gives the number of jobs and N gives the number of machines. LB stands for the lower bound values obtained for the benchmark problems.

RESULTS: WITH ANTIBODY POPULATION 30 AND CLONE POPULATION 100

PROBLEM	M	N	LB	Makespan(AIS)	%RE
Ft06	6	6	55	55	0
Ft10	10	10	930	930	0
La01	10	5	666	666	0
La02	10	5	655	655	0
La03	10	5	597	597	0
La04	10	5	590	590	0
La05	10	5	593	593	0
La06	15	5	926	926	0
La08	15	5	863	863	0
La09	15	5	951	959	0.84
La10	15	5	958	958	0
La11	20	5	1222	1222	0
La12	20	5	1039	1039	0
La13	20	5	1150	1150	0
La14	20	5	1292	1292	0
La15	20	5	1207	1213	0.49
ORB01	10	10	1059	1071	1.13
ORB02	10	10	888	894	0.67
ORB03	10	10	1005	1005	0
ORB04	10	10	1005	1005	0
ORB05	10	10	887	887	0
ORB06	10	10	1010	1028	1.78
ORB07	10	10	397	397	0
ORB08	10	10	899	899	0
ORB09	10	10	934	934	0
ORB10	10	10	944	944	0

TABLE 3: RESULTS OBTAINED BY AIS PROCEDURE.

The makespan values obtained are also compared with the make span values obtained by PSO [25]procedure and it shows that in many cases a better result is obtained in AIS procedure.

The makespan values were also compared with the number of iterations. The convergence of makespan with the number of iterations was also plotted in a graph

Figure 5:COMPARISON OF RESULTS OBATINED IN AIS VS TSSB

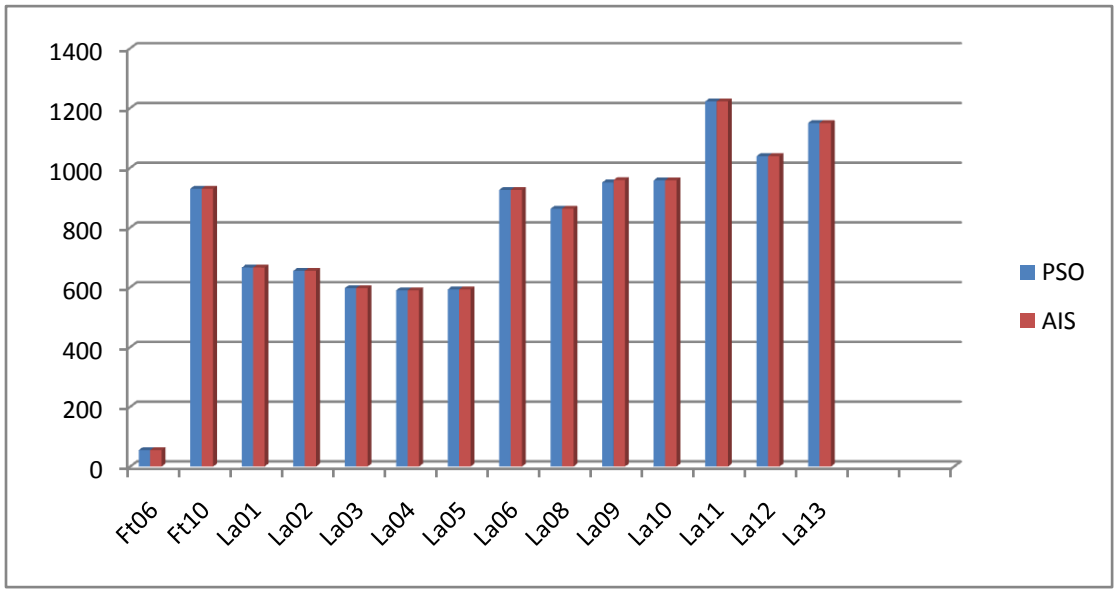


Figure 6: Comparison of Results obtained in AIS with TSSB

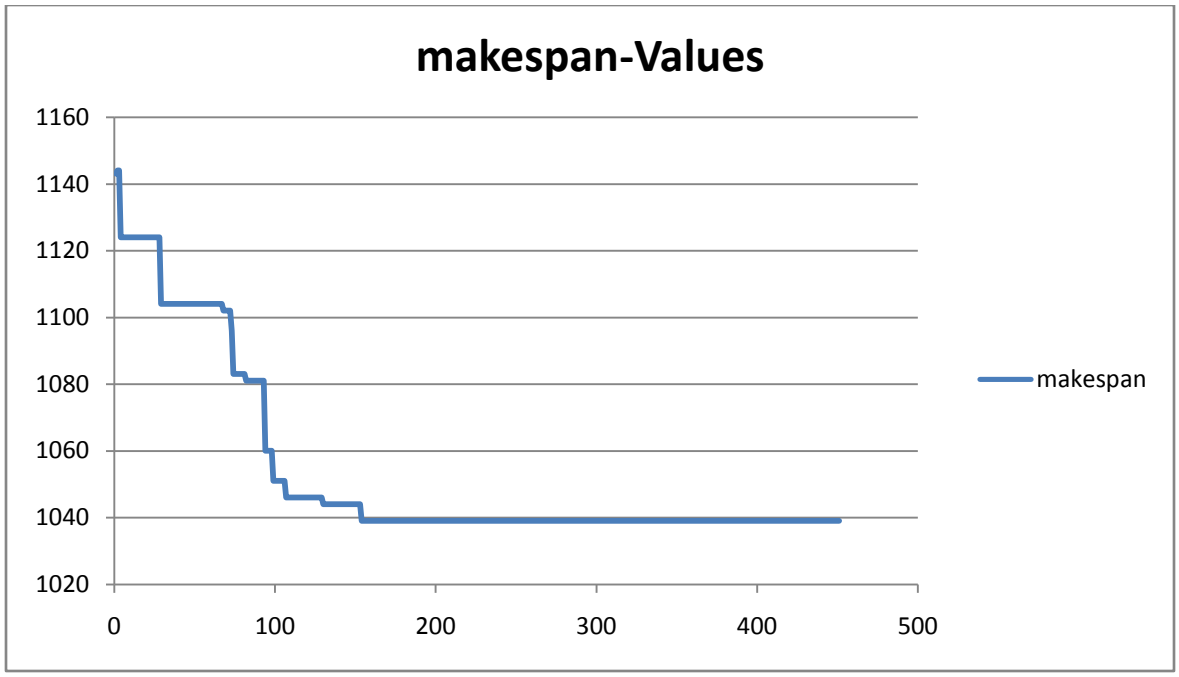


Figure 7:GRAPH OF MAKESPAN VS THE NO OF ITERATIONS FOR THE PROBLEM LA12

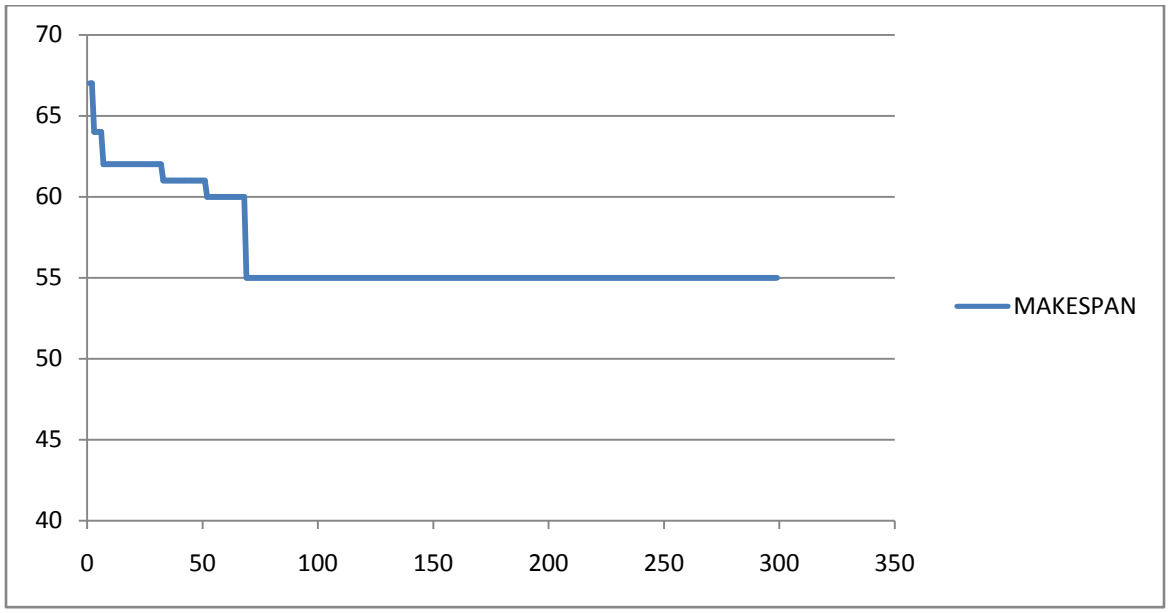


Figure 8: Graph of makepsanvs number of iterations for the problem Ft06.

The graph shows that after a certain number of iterations the makespan values converge rapidly.

After a number of iterations when the makespan value doesn't change anymore the optimal solution is obtained.

CONCLUSION

The present work is focused on Scheduling of jobs in Job shop using artificial immune system. Job shop scheduling using AIS aims at minimizing the makespan time. The algorithm uses simple inverse and pair wise exchange mutation and a receptor editing process. The algorithm has been encoded in devC++. The algorithm proved to be efficient in many of the benchmark problems. The schedules obtained have makepsan value near to optimal.

REFERENCES

REFERENCES:

1. Bruker P (1995) Scheduling algorithms 2nd edn. Springer, Berlin Heidelberg New York.

2. Garey M, et al (1976) The complexity of flow shop and Job shop scheduling, MathematicalOperation Research 1:117-119.

3. Brandimarte P. (1993) routing and scheduling in a flexible job shop by taboo search, Annals of operation research, 41, 157-183

4. Carlier J, Pison E (1989) An algorithm for solving Job shop problem. Management Science 35: 164-176

5. Bruker P et al (1994) A branch and bound algorithm for Jobshop problem. Discrete Applied Mathematics 49:107-127

6. Blazewickz J (1996) The Job shop scheduling Problem : Conventional and new solutions techniques. European Journal of Operational Researchpp 931-30

7. Erscher JF, Roubellat JP, Verntes (1976) Finding some essential characteristics of the feasible solutions for a scheduling problem. Operational research: 24:774-783

8. Heurik .E, Jarisch B. and Thole M. (1994) Tabu search for the job shop scheduling problem with multipurpose machines. Operation research Spectrum, 15 205-215

9. MastrolliM.and Gambardella C M (2000) Effective neighborhood functions for the flexible jobshop problem. Journal of scheduling 3(1) ,3-20

10. R. Tavakkoli-Moghaddam, F. Jolai, F. Vaziri, P.K. Ahmed, A. Azaron., A hybrid method for solving stochastic job shop scheduling problem.Applied Mathematics and computation 170 (2005)185-206.

11. Daeyoung Chung, Kichang Lee, Kitae Shin, Jinwoo Park. A new approach to jobshop scheduling problems with due date constraints considering operation subcontracts. International Journal of Production Economics 98 (2005) 238–250

12. D.L. Santos, J.L. Hunsucker, D.E. Deal, Global lower bounds for flow shops with multiple processors.European Journal of Operational Research 80(1995) 112-130

13. JinweiGu, Xingsheng Gu, Manzhan Gu., A novel parallel quantum genetic algorithm for stochastic Job shop scheduling. Journal of Mathematical Analysis and Applications 355(2009)63-81

14. QuinNiu, Bin Jiao, Xingsheng Gu., Particle Swarm optimization combined with genetic operators for job shop scheduling problem with fuzzy processing time.Applied Mathematics and Computation 205 (2008) 148-158.

15. J Timmisa, A. Honec, T. Stibord, E. Clarka. Theoretical advances in artificial immune systems.Theoretical Computer Science 403(2008)11-32

16. JieGao, Linyan Sun, Mitsuo Gen.A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problem. Computers and Operation Research 35(2008) 2892-2907

17. Q Zhang, H Manier, M.A. Manier.A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraint and bounded processing times. Computers and Operation Research 39(2012) 1713-1723

18. EbruDemirkol, Sanjay Mehta, RehaUzosy. Benchmarks for Job shop scheduling problems.European Journal of Operational Research 109 (1998) 137-141

19. Lawrence S (1984) Supplement to resource constrained project scheduling: an experimental investigation of heuristic schedulingtechniques. TechnicalRepresenation, GSIA, Carnegie Mellon University.

20. Applegate D, Cook W (1991) A computational study of the job shop scheduling problem. ORSA Journal of Computing 3:149–156

21. Engin O, Doyen A (2004) A new approach to solve hybrid flow shop scheduling problems by artificial immune system. Future Generation Computer Systems 20:1083–1095

22. M.Chandrasekaran,P.Asokan,S.Kumanan,T.Balamurugan,S.Nickolas Solving job shop scheduling problemsusing artificial immune system. International Journal of Advanced Manufacturing Technology (2006) 31: 580–593

23. OrhanEngin, AlperDöyen. A new approach to solve hybrid flow shop scheduling problems by artificial immune system. Future Generation Computer Systems 20 (2004) 1083–1095

24. Runwei Cheng, Mitsuo Gen and Yasuhiro Tsujimura. A tutorial Survey of Jobshop Scheduling Problems using genetic algorithms-I. Representation. Computers industrial Engineering Vol 30 No 4 pp 983-997, 1996

25. Anan Banharnsakun,BooncharoenSirinaovakul,TiraneeAchalakul. Job ShopScheduling with the Best-so-far ABC. Engineering Applications of Artificial Intelligence25(2012)583–593