Int J Adv Manuf Technol (2006) 31: 580–593
DOI 10.1007/s00170-005-0226-3

ORIGINAL ARTICLE

M. Chandrasekaran · P. Asokan · S. Kumanan · T. Balamurugan · S. Nickolas

Solving job shop scheduling problems using artificial immune system

Received: 8 February 2005 / Accepted: 30 July 2005 / Published online: 3 January 2006

Abstract The n-job, m-machine job shop scheduling (JSS) problem is one of the general production scheduling problems. Many existing heuristics give solutions for small size problems with near optimal solutions. This paper deals with the criterion of makespan minimization for the job shop scheduling of different size problems. The proposed computational method of artificial immune system algorithm (AIS) is used for finding optimal makespan values of different size problems. The artificial immune system algorithm is tested with 130 benchmark problems [10 (ORB1-ORB5 & ARZ5-ARZ9), 40 (LA01-LA40) and 80 (TA01-TA80)]. The results show that the AIS algorithm is an efficient and effective algorithm which gives better results than the Tabu search shifting bottleneck procedure (TSSB) as well as the best solution of shifting bottleneck procedure (SB-GLS1) of Balas and Vazacopoulos.

Keywords Affinity mutation · Artificial immune system · Clonal selection · Job shop scheduling

Notations used

AIS	Best solution of AIS algorithm
m	Number of machines
MRE	Mean relative error in percent for a set of problems
n	Number of jobs
Opt (LB UB)	The optimal value of known best lower and Upper bound, from OR-Library
$RE_{SB\text{-}GLS1}$	Percent relative error by SB-GLS1
RE_{TSSB}	Percent relative error by TSSB
RE_{AIS}	Percent relative error by AIS
RE	Relative error in percent
SB-GLS1	Best solution of SB-GLS1 procedure of Balas and Vazacopoulos
TSSB	Best solution of TSSB
Tav	Average computing time in seconds

M. Chandrasekaran (✉) · P. Asokan · S. Kumanan · T. Balamurugan
Department of Production Engineering, National Institute of Technology, Tiruchirappalli, 620015 Tamilnadu, India
e-mail: ch_sekaran@yahoo.com

S. Nickolas
Department of Computer Applications, National Institute of Technology, Tiruchirappalli, 620015 Tamilnadu, India

1 Introduction

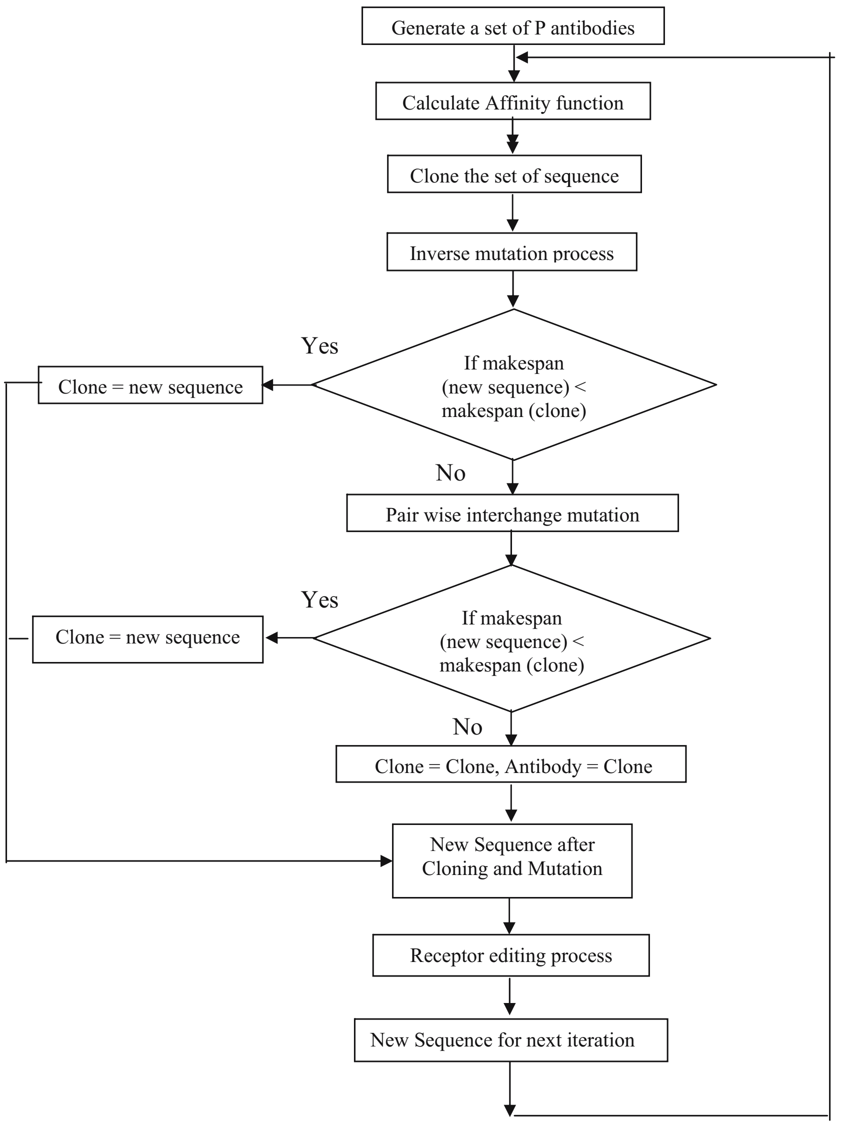
A schedule is an allocation of the tasks to time intervals on the machines and the aim is to find a schedule that minimizes the overall completion time, which is called the makespan. In the job shop scheduling problem n jobs have to be processed on m different machines. Each job consists of a sequence of tasks that have to be processed during an uninterrupted time period of a fixed length on a given machine. So the maximum of completion times needed for processing all jobs, subject to the constraints that each job has a specified processing order through the machines and that each machine can process at most one job at a time.

Bruker [1] and Garey [2] show that the job shop scheduling is an NP-hard combinatorial problem. Because of the NP-hard characteristics of job shop scheduling, it is usually very hard to find its optimal solution, and an optimal solution in the mathematical sense is not always necessary in practice [3]. Researchers turned to search its near-optimal solutions with all kinds of heuristic algorithms [4]. Fortunately, the searched near optimal solutions usually meet requirements of practical problems very well.

Carlier and Pison [5], Bruker [6] managed to generate optimal schedules using an algorithm and branch and bound approach, but only for small size problems. Therefore the effort has been directed by Blazewicz [7] to the development of efficient local search based heuristics to solve practical size problems.

Tabu search with bottleneck procedure (TSSB) was used to test the job shop scheduling benchmark problem instances [8] in the past work. The outcome of that TSSB procedure was compared with literature results [9]. It gave an op-

Fig. 1 Flow chart of artificial immune system algorithm for job shop scheduling


Generate a set of P antibodies
Calculate Affinity function
Clone the set of sequence
Inverse mutation process
If makespan (new sequence) < makespan (clone)
Yes
Clone = new sequence
No
Pair wise interchange mutation
If makespan (new sequence) < makespan (clone)
Yes
Clone = new sequence
No
Clone = Clone, Antibody = Clone
New Sequence after Cloning and Mutation
Receptor editing process
New Sequence for next iteration

timum bound value for a moderate number of problems and with considerable amount of relative error. In this work an attempt is made to produce an optimum bound value for the maximum number of problems and to minimize the mean relative error with this new AIS algorithm.

The AIS algorithm is already used in different applications namely computer and network security [10, 11], fault and anomaly detection [12, 13], optimization [14, 15], data analysis and data mining [16, 17] and flow shop scheduling where AIS produced better results.

Many authors have been trying to bring out the utility and advantages of heuristics and other evolutionary algorithms. It is proposed to use new evolutionary concept viz. the artificial immune system for solving job shop scheduling problems.

1.1 Job shop scheduling problem

Normally, the entire job shop scheduling problem consists of two types of constraints: sequence constraint and resource constraint [18]. The first type states that two operations of a job cannot be processed at the same time.

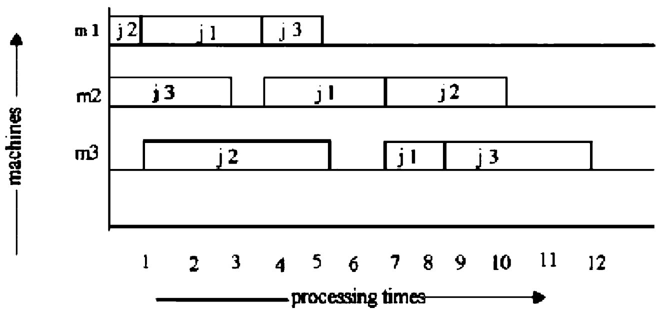


Fig. 2 Feasible schedule

Table 1 Operational sequences for each job

8	12	9	4	13	2	14	1	15	7	10	5	3	11	6
13	2	12	10	7	4	3	5	6	9	14	15	11	1	8
2	3	10	1	4	6	9	5	15	11	13	14	8	7	12
14	11	7	3	15	8	5	12	1	6	10	4	9	2	13
2	9	5	15	7	6	4	3	10	11	14	8	12	1	13
6	15	3	13	11	2	12	5	7	10	1	14	9	4	8
6	3	1	2	9	15	12	11	8	10	7	13	5	14	4
5	8	11	2	10	9	3	15	12	4	6	7	14	13	1
15	12	1	10	11	6	4	13	9	14	7	2	8	3	5
10	1	4	11	13	14	6	2	7	15	9	12	3	8	5
8	3	2	13	4	15	5	7	6	10	9	14	11	1	12
1	9	15	13	10	6	7	11	8	12	4	5	2	14	3
9	13	11	12	15	4	7	2	5	6	1	10	14	3	8
14	3	12	1	15	11	4	2	13	5	6	7	8	10	9
2	14	1	12	3	11	5	9	4	6	8	7	10	13	15

The second type states that no more than one job can be handled on a machine at the same time. Job shop scheduling can be viewed as an optimization problem, bounded by both sequence and resource constraints. For a job shop scheduling problem, each job may consist of a different number of operations, subjected to some precedence restrictions. Commonly the processing orders of each job by all machines and the processing time of each operation are known and fixed. Once started operations cannot be interrupted.

Assume job i (i=1,2,...n) requires processing by machine k (k=1,2,... m) exactly once in its operation sequence (thus, each job has m operations). Let p_{ik} be the processing time of job i on machine k, X_{ik} be the starting time of job i on machine k, q_{ijk} be the indicator which takes on a value of 1 if operation j of job i requires machine k, and zero otherwise. Y_{ihk} is the variable which takes on a value of 1 if job i precedes job h on machine k, and zero otherwise. The objective function for the given job shop scheduling is

$$\text{Minimize } Z = \sum_{k=1}^{m} q_{imk}(X_{ik} + p_{ik}) \qquad (i = 1, ...n)$$

Subject to

(a) Sequence constraint

$$\sum_{k=1}^{m} q_{ijk}(X_{ik} + p_{ik}) \leq \sum_{k=1}^{m} q_{i,j+1,k} X_{ik}, \quad (i = 1, ...n;\ j = 1, ...m - 1)$$

ie., for a given job i, the $(j+1)^{st}$ operation may not start before the j^{th} operation is completed.

Table 2 Processing times of each job

69	81	81	62	80	3	38	62	54	66	88	82	3	12	88
83	51	47	15	89	76	52	18	22	85	26	30	5	89	22
62	47	93	54	38	78	71	96	19	33	44	71	90	9	21
33	82	80	30	96	31	11	26	41	55	12	10	92	3	75
36	49	10	43	69	72	19	65	37	57	32	11	73	89	12
83	32	6	13	87	94	36	76	46	30	56	62	32	52	72
29	78	21	27	17	43	14	15	16	49	72	19	99	38	64
12	74	4	3	15	62	50	38	49	25	18	55	5	71	27
69	13	33	47	86	31	97	48	25	40	94	22	61	59	16
27	4	35	80	49	46	84	46	96	72	18	23	96	74	23
36	17	81	67	47	5	51	23	82	35	96	7	54	92	38
78	58	62	43	1	56	76	49	80	26	79	9	24	24	42
38	86	38	38	83	36	11	17	99	14	57	64	58	96	17
10	86	93	63	61	62	75	90	40	77	8	27	96	69	64
73	12	14	71	3	47	84	84	53	58	95	87	90	68	75

Table 3 Randomly generated job sequences

Seq. No		Generated job sequences														
1	–>	15	1	5	12	10	9	4	7	13	2	14	8	6	3	11
2	–>	14	8	3	10	5	2	1	7	6	9	15	4	12	11	13
3	–>	12	7	15	10	5	8	13	3	14	6	2	4	9	1	11
4	–>	4	8	1	14	12	6	7	10	5	13	3	9	15	11	2
5	–>	7	14	12	10	4	8	15	13	1	2	6	5	11	3	9
6	–>	13	11	10	5	15	9	2	3	7	4	8	12	6	14	1
7	–>	5	8	12	7	1	9	10	4	6	2	14	15	11	13	3
8	–>	10	13	9	1	4	3	2	15	5	8	6	11	14	12	7
9	–>	4	13	10	1	5	14	11	3	7	6	2	9	8	12	15
10	–>	2	3	12	10	11	6	15	7	14	5	9	13	4	8	1

(b) Resource constraint

$$X_{hk} - X_{ik} \geq p_{ik} - (H + p_{ik})(1 - Y_{ihk}),$$
$$X_{ik} - X_{hk} \geq p_{hk} - (H + p_{hk})\ Y_{ihk},$$

where $(i = 1,.....n;\ h = 1,.....n;\ k = 1,......m)$
where H is a very large positive integer, chosen so that only one of the above constraints binding either for $Y_{ihk} = 1$ or for $Y_{ihk} = 0$.

1.2 Artificial immune system (AIS)

The operative mechanisms of immune system are very efficient from a computational standpoint [19]. The artificial immune system was built on the following two principles of the immune system.

(a) Clonal selection principle
(b) Affinity maturation principle

1.2.1 Cloning selection principle

Each schedule (antibody) has a makespan value that refers to the affinity value of that antibody. Affinity value of each schedule is calculated from the affinity function. The affinity function is defined as

$$\text{Affinity (p)} = \frac{1}{\text{makespan.}}$$

From this relation, a lower makespan value gives a higher affinity value. Further the cloning of antibodies is done directly proportional to their affinity function values. Therefore, there will be more clones of antibodies that have lower makespan values than those with higher makespan values in the new generated clone population. An immune-based approach to minimize makespan on parallel processors has been presented in [20]. An affinity function is defined based on makespan values of the schedules [21, 22]. Also they have given a function to calculate the number of clones [23, 24] that would be proliferated.

1.2.2 Affinity maturation principle

The affinity maturation principle consists of two methods namely mutation and receptor editing.

Table 4 Evaluation results of makespan and affinity for each job sequence

Seq. No		Job sequences															Makespan	Affinity
1	–>	15	1	5	12	10	9	4	7	13	2	14	8	6	3	11	1396	0.000716
2	–>	14	8	3	10	5	2	1	7	6	9	15	4	12	11	13	1589	0.000629
3	–>	12	7	15	10	5	8	13	3	14	6	2	4	9	1	11	1348	0.000742
4	–>	4	8	1	14	12	6	7	10	5	13	3	9	15	11	2	1376	0.000727
5	–>	7	14	12	10	4	8	15	13	1	2	6	5	11	3	9	1476	0.000678
6	–>	13	11	10	5	15	9	2	3	7	4	8	12	6	14	1	1270	0.000787
7	–>	5	8	12	7	1	9	10	4	6	2	14	15	11	13	3	1407	0.000711
8	–>	10	13	9	1	4	3	2	15	5	8	6	11	14	12	7	1567	0.000638
9	–>	4	13	10	1	5	14	11	3	7	6	2	9	8	12	15	1471	0.00068
10	–>	2	3	12	10	11	6	15	7	14	5	9	13	4	8	1	1238	0.000808
Summation of all affinity values																		0.007116

Table 5 Cloning process in proposed AIS

Seq. No	No.of clones	Job sequences after cloning process															Makespan
1	2	15	1	5	12	10	9	4	7	13	2	14	8	6	3	11	1396
		15	1	5	12	10	9	4	7	13	2	14	8	6	3	11	1396
2	1	14	8	3	10	5	2	1	7	6	9	15	4	12	11	13	1589
3	2	12	7	15	10	5	8	13	3	14	6	2	4	9	1	11	1348
		12	7	15	10	5	8	13	3	14	6	2	4	9	1	11	1348
4	2	4	8	1	14	12	6	7	10	5	13	3	9	15	11	2	1376
		4	8	1	14	12	6	7	10	5	13	3	9	15	11	2	1376
5	1	7	14	12	10	4	8	15	13	1	2	6	5	11	3	9	1476
6	2	13	11	10	5	15	9	2	3	7	4	8	12	6	14	1	1270
		13	11	10	5	15	9	2	3	7	4	8	12	6	14	1	1270
7	1	5	8	12	7	1	9	10	4	6	2	14	15	11	13	3	1407
8	1	10	13	9	1	4	3	2	15	5	8	6	11	14	12	7	1567
9	1	4	13	10	1	5	14	11	3	7	6	2	9	8	12	15	1471
10	2	2	3	12	10	11	6	15	7	14	5	9	13	4	8	1	1238
		2	3	12	10	11	6	15	7	14	5	9	13	4	8	1	1238

Mutation A two phased mutation procedure were used for the generated clones [25].

(a) Inverse mutation
(b) Pairwise interchange mutation

(a) Inverse mutation: For a sequence s, let i and j be randomly selected two positions in the sequences. A neighbor of s is obtained by inversing the sequence of jobs between i and j positions. If the makespan value of the mutated sequence (after inverse mutation) is smaller than that of the original sequence (a generated clone from an antibody), then the mutated one is stored in the place of the original one. Otherwise, the sequence will be mutated again with random pair wise interchange mutation.

(b) Pair wise interchange mutation: Given a sequence s, let i and j be randomly selected two positions in the sequence s. A neighbor of s is obtained by interchanging the jobs in positions i and j. If the makespan value of the mutated sequence (after pair wise interchange mutation) is smaller than that of the original sequence, then store the mutated one in the place of the original one. In the case where the algorithm could not find a better sequence after the two-mutation procedure, then it stores the original sequence (generated clone).

Receptor editing After cloning and mutation processes, a percentage of the antibodies (worst %B of the whole population) in the antibody population are eliminated and randomly created antibodies are replaced with them. This mechanism allows to find new schedules that correspond to new search regions in the total search space [26].

Table 6 Mutation process in proposed AIS

Seq. No	No. of clones	Makespan original	Inverse mutation makespan	Pairwise mutation makespan	Selected job sequence after mutation															Make span
1	2	1396	1420	1397	15	1	5	12	10	9	4	7	13	2	14	8	6	3	11	1396
		1396	1319	1464	7	4	9	10	12	5	1	15	13	2	14	8	6	11	3	1319
2	1	1589	1589	1493	13	11	12	4	15	1	7	6	9	2	5	10	3	8	14	1493
3	2	1348	1332	1460	8	5	10	15	7	12	13	3	14	6	11	1	9	4	2	1332
		1348	1348	1490	12	7	15	10	5	8	13	3	14	6	2	4	9	1	11	1348
4	2	1376	1420	1255	2	11	15	10	5	13	3	9	7	6	12	14	1	8	4	1255
		1376	1388	1255	2	11	15	9	3	10	5	13	7	6	12	14	1	8	4	1255
5	1	1476	1397	1405	12	14	7	10	4	8	9	3	11	5	6	2	1	13	15	1397
6	2	1270	1262	1338	4	7	3	2	9	15	5	10	11	13	8	12	6	1	14	1262
		1270	1405	1376	13	11	10	5	15	9	2	3	7	4	8	12	6	14	1	1270
7	1	1407	1407	1319	3	13	11	12	7	1	9	10	4	6	2	14	15	8	5	1319
8	1	1567	1512	1303	4	1	9	13	10	3	2	15	5	8	6	11	14	7	12	1512
9	1	1471	1301	1375	3	11	14	5	1	10	13	4	7	6	2	9	15	12	8	1301
10	2	1238	1331	1476	2	3	12	10	11	6	15	7	14	5	9	13	4	8	1	1238
		1238	1391	1378	2	3	12	10	11	6	15	7	14	5	9	13	4	8	1	1238

Table 7 Receptor editing process in proposed AIS

Seq. No		Job sequences after receptor editing process														
1	->	2	3	12	10	11	6	15	7	14	5	9	13	4	8	1
2	->	2	11	15	10	5	13	3	9	7	6	12	14	1	8	4
3	->	2	11	15	9	3	10	5	13	7	6	12	14	1	8	4
4	->	4	7	3	2	9	15	5	10	11	13	8	12	6	1	14
5	->	13	11	10	5	15	9	2	3	7	4	8	12	6	14	1
6	->	3	11	14	5	1	10	13	4	7	6	2	9	15	12	8
7	->	3	13	11	12	7	1	9	10	4	6	2	14	15	8	5
8	->	**6**	**9**	**7**	**4**	**5**	**3**	**11**	**2**	**10**	**13**	**15**	**1**	**8**	**12**	**14**
9	->	**1**	**7**	**4**	**2**	**6**	**10**	**5**	**11**	**13**	**3**	**8**	**9**	**15**	**12**	**14**
10	->	**3**	**6**	**11**	**1**	**14**	**7**	**15**	**9**	**2**	**4**	**12**	**5**	**13**	**8**	**10**

1.3 Artificial immune system vs. genetic algorithms (GA)

Both AIS and GA are population based evolutionary and biologically inspired algorithms. There is no difference between a chromosome in GA and an antibody in the AIS, thus they do not differ in representation of the components. Also, GA have fitness function and, similarly, AIS have an affinity function to evaluate the quality of each individual. However, their evolutionary search procedures differ from each other.

Here, we point out the difference between our AIS approach and a GA. First of all, reproduction and selection of new generations are different. In a standard GA, crossover and mutation are the basic tools for creating new solutions, in AIS it is achieved by cloning and mutating antibodies due to their affinity function values. In AIS, mutations are immediately tested for acceptance or rejection, rather than a phase where a new generation is selected as in GA [19].

This paper is organized as follows. Section 2 deals with the proposed AIS algorithm for the job shop scheduling problem. Section 3 presents the implementation of AIS with a numerical illustration. Section 4 deals the results and discussion of 130 benchmark test problems with literature results. Finally, the conclusion of the research work is presented in Sect. 5.

2 Proposed AIS algorithm for job shop scheduling problem

The flow chart in Fig. 1 gives the details of AIS algorithm for solving the job shop scheduling problem. In this work, possible schedules are represented by integer-valued sequences of length n (jobs). The n elements of the strings are the jobs which will be sequenced. Therefore, the strings are composed of permutations of n (jobs) elements. Those strings are accepted as antibodies of the AIS. The algorithm goes up to solution by the evolution of these antibodies.

```
Generate a population of P antibodies (job sequences).
For each iteration:
   Select the sequence in the antibody population;
   Find out the affinity of each antibody;
   Cloning process (generate copies of the antibodies).
Steps in mutation process (for each clone)
   Find inverse mutation (generate a new sequence):
   Select the new sequence obtained from inverse mutation:
   Find the makespan of the new sequence:
   if makespan (new sequence) < makespan (clone) then
   Clone = new sequence
   else,
   do pair wise interchange mutation (generate a new sequence):
```

Table 8 Final results obtained after 1,500 iterations in proposed AIS

Seq. No		Final job sequences															Makespan
1	->	1	2	4	6	10	3	11	15	5	12	7	14	13	9	8	1206
2	->	6	1	2	4	10	3	11	15	5	12	7	14	13	9	8	1206
3	->	2	4	1	6	10	3	11	15	5	12	7	14	13	9	8	1206
4	->	5	13	11	6	2	10	15	14	1	12	4	7	3	8	9	1206
5	->	10	5	13	15	6	14	1	2	3	12	9	4	11	7	8	1206
6	->	4	2	1	6	10	3	11	15	5	12	7	14	13	8	9	1206
7	->	6	3	2	5	10	11	15	14	4	12	13	9	7	8	1	1206
8	->	6	13	5	3	4	11	9	8	10	15	2	14	12	7	1	1208
9	->	11	6	13	5	3	4	9	8	10	15	2	14	12	7	1	1208
10	->	12	13	4	14	10	11	15	8	2	5	6	7	1	3	9	1209

Table 9 Results for ten instances (ORB1-ORB5) and (ABZ5-ABZ9) of class (i) problems

Problem	n	m	Opt (LB UB)	**AIS**	**RE$_{AIS}$**	**CPU Time sec.**	TSSB	RE$_{TSSB}$	**CPU Time sec.**
ORB1	10	10	1059	**1062**	**0.22**	**96**	1064	0.47	82
ORB2	10	10	888	**891**	**0.34**	**93**	890	0.23	75
ORB3	10	10	1005	**1005**	**0**	**28**	1013	0.8	87
ORB4	10	10	1005	**1005**	**0**	**35**	1013	0.8	75
ORB5	10	10	887	**889**	**0.23**	**98**	887	0	81
ABZ5	10	10	1234	**1234**	**0**	**32**	1234	0	75
ABZ6	10	10	943	**943**	**0**	**37**	943	0	80
ABZ7	20	15	656	**666**	**1.52**	**256**	666	1.52	200
ABZ8	20	15	(645 669)	**669**	**0**	**118**	678	5.12	205
ABZ9	20	15	(661 679)	**684**	**0.74**	**242**	693	4.84	195
Mean Relative Error (MRE)					**0.305**			1.378	

```
    select the new sequence:
    Find the makespan of the new sequence:
    If makespan (new sequence) < makespan (clone) then
    clone = new sequence:
    else
    clone = clone:
    antibody = clone:
Eliminate worst %B number of antibodies in the population:
    Create new antibodies at the same number (%B of pop.)
    Change the eliminated ones with the new created ones
    while stopping criteria = false.
```

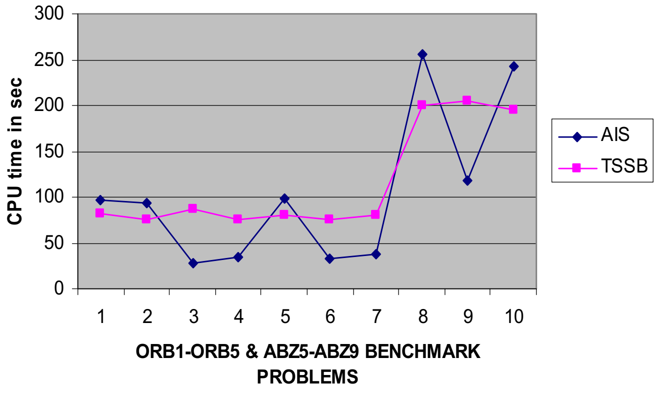


Fig. 3 The graph shows that CPU times obtained for class (i) problems in AIS compared with TSSB procedure

3 Implementation of AIS

3.1 Software development

The artificial immune system algorithm is implemented in C language on a personal computer Pentium IV 2.4 GHz. The maximum number of iterations has been set to 100 X n, where n is the number of jobs.

Fig. 4 The graph shows that makespan results obtained for class (i) problems in AIS compared with TSSB procedure

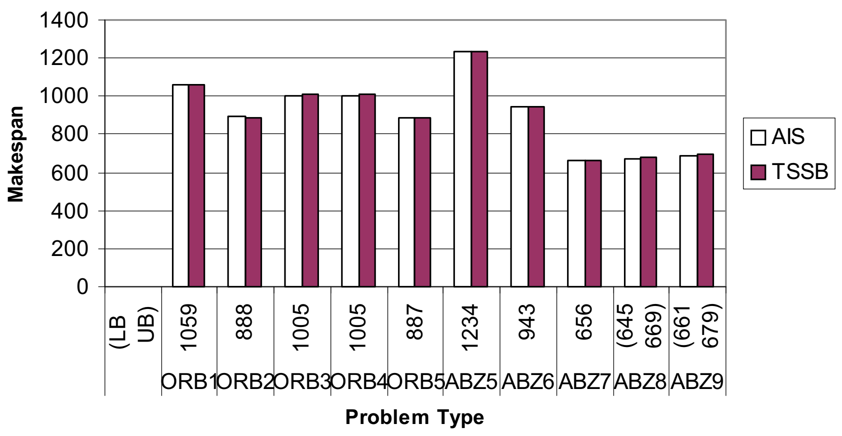


Table 10 Results for forty instances (LA01-LA40) of class (ii) problems

Problem	n	m	Opt (LB UB)	**AIS**	**RE$_{AIS}$**	TSSB	RE$_{TSSB}$
LA01	10	5	666	**666**	**0**	666	0
LA02	10	5	655	**655**	**0**	655	0
LA03	10	5	597	**601**	**0.67**	597	0
LA04	10	5	590	**590**	**0**	590	0
LA05	10	5	593	**593**	**0**	593	0
LA06	15	5	926	**926**	**0**	926	0
LA07	15	5	890	**890**	**0**	890	0
LA08	15	5	863	**863**	**0**	863	0
LA09	15	5	951	**951**	**0**	951	0
LA10	15	5	958	**958**	**0**	958	0
LA11	20	5	1222	**1222**	**0**	1222	0
LA12	20	5	1039	**1039**	**0**	1069	0
LA13	20	5	1150	**1150**	**0**	1150	0
LA14	20	5	1292	**1292**	**0**	1292	0
LA15	20	5	1207	**1207**	**0**	1207	0
LA16	10	10	945	**945**	**0**	945	0
LA17	10	10	784	**784**	**0**	784	0
LA18	10	10	848	**848**	**0**	848	0
LA19	10	10	842	**842**	**0**	842	0
LA20	10	10	902	**902**	**0**	902	0
LA21	15	10	1046	**1046**	**0**	1046	0
LA22	15	10	927	**929**	**0.21**	927	0
LA23	15	10	1032	**1032**	**0**	1032	0
LA24	15	10	935	**935**	**0**	938	0.32
LA25	15	10	977	**977**	**0**	979	0.2
LA26	20	10	1218	**1218**	**0**	1218	0
LA27	20	10	1235	**1235**	**0**	1235	0
LA28	20	10	1216	**1216**	**0**	1216	0
LA29	20	10	(1142 1153)	**1153**	**0**	1168	2.28
LA30	20	10	1355	**1355**	**0**	1355	0
LA31	30	10	1784	**1784**	**0**	1784	0
LA32	30	10	1850	**1850**	**0**	1850	0
LA33	30	10	1719	**1719**	**0**	1719	0
LA34	30	10	1721	**1721**	**0**	1721	0
LA35	30	10	1888	**1888**	**0**	1888	0
LA36	15	15	1268	**1273**	**0.39**	1268	0
LA37	15	15	1397	**1397**	**0**	1411	1
LA38	15	15	1196	**1196**	**0**	1201	0.42
LA39	15	15	1233	**1233**	**0**	1240	0.57
LA40	15	15	1222	**1230**	**0.65**	1233	0.9
Mean Relative Error (MRE)					**0.048**		0.14

3.2 Representation of solution seed

Consider the three-job three-machine problem shown below.

Processing time operations				Machine sequence operations			
Job	1	2	3	Job	1	2	3
J1	3	3	2	j1	m1	m2	m3
j2	1	5	3	j2	m1	m3	m2
j3	3	2	3	j3	m2	m1	m3

Suppose a seed is given as [3 2 1], where 1 stands for job j1, 2 for job j2, and 3 for job j3. This sequence has to be operated 3 times in the same order because each job has three operations. So, we can consider the initial seed as the following format [3 2 1 3 2 1 3 2 1]. There are three 2s in the seed, which stands for the three operations of job j2. The first 2 corresponds to the first operation of job j2 which will be processed on machine 1, the second 2 corresponds to the second operation of job j2 which will be processed on machine 3, and the third 2 corresponds to the third operation of job j2 which will be processed on machine 2.

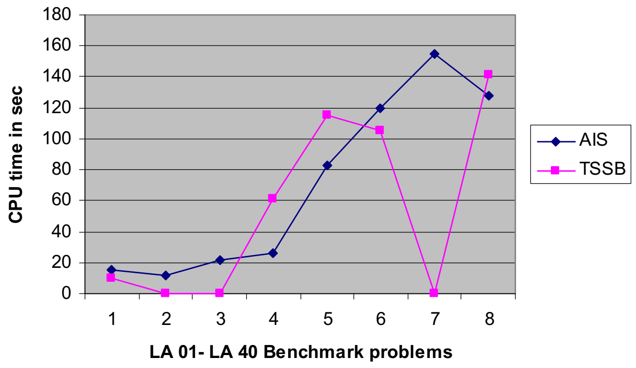


Fig. 5 The graph shows that CPU times obtained for class (ii) problems in AIS and TSSB procedure

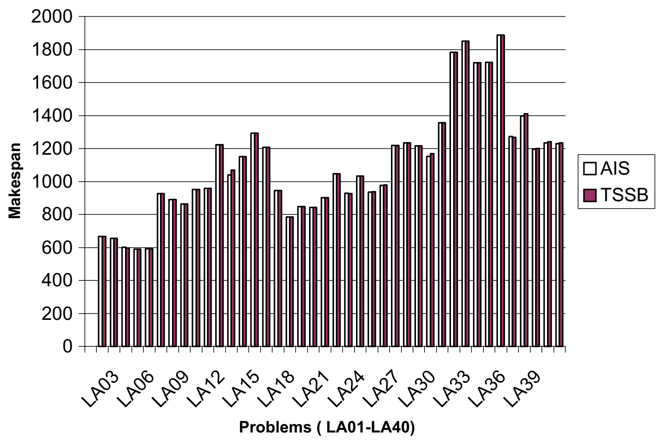


Fig. 6 The graph shows that results obtained for class (ii) problems in AIS and TSSB procedure

We can see that all operations for job j2 are given the same symbol 2 and then interpreted according to their orders of occurrence in the sequence of this seed. The corresponding relationships of the operations of jobs and processing machines are shown in Fig. 2.

This concept is used to find the makespan for the sequences of the benchmark problems where the generated seed (job sequence) is operated equal to the number of machines represented in the particular problem.

3.3 Numerical example

Input data TYPE 15x15 - (TA 03), Number of machines = 15, Number of jobs = 15

The operational sequences and processing times are given in Tables 1 and 2.

Initialization In the initialization phase, job sequences are generated randomly for 15 X 15 (TA-03) problem, which is given in Table 3.

Evaluation For each sequence makespan and its affinity values are calculated. Affinity is calculated as affinity (p) = 1/ make span. Finally summation of all affinity values of a given population set are also calculated and shown in Table 4.

Cloning process Based on the affinity value of the individual sequence number of clones are calculated for the population.

Number of clones = Round of [(individual affinity/total affinity)×Population size].

Population size = 10 and is shown in Table 5.

Mutation process In the mutation process, if original sequence makespan is less than the makespan obtained after inverse mutation and pairwise interchange mutation, then the original sequence is retained. Otherwise, the sequence generated during inverse or pairwise interchange mutation is taken based on mutation condition, which is shown in Table 6.

Receptor editing After sorting the sequence and deleting repetition, the receptor editing process has to be done. Of the worst sequences, 30% are replaced by new randomly generated sequences and are shown in Table 7. This gives the possibility of a new search space in the total search space. These sequences will be taken as input for the next

Table 11 Comparison of mean relative error and computing time of AIS with TSSB [8]

Problem	n	m	AIS		TSSB	
			MRE	Tav	MRE	Tav
LA 01–05	10	5	0.12	15	0.00	9.8
LA 06–10	15	5	0	12	0.00	–
LA 11–15	20	5	0	22	0.00	–
LA 16–20	10	10	0	26	0.00	61.5
LA 21–25	15	10	0.04	83	0.10	115
LA 26–30	20	10	0	120	0.46	105
LA 31–35	30	10	0	155	0.00	–
LA 36–40	15	15	0.21	128	0.58	141
			0.048		0.14	

Table 12 Results of 80 instances of eight different size (n×m = 15×15; 20×15; 20×20; 30×15; 30×20; 50×15; 50×20; 100×20) of class (iii) problems

Problem	n	m	Opt (LB UB)	**AIS**	**RE_{AIS}**	TSSB	RE_{TSSB}	SB-GLS1	$RE_{SB\text{-}GLS1}$
TA1	15	15	1231	**1231**	**0**	1241	0.812	1244	1.056
TA2	15	15	1244	**1244**	**0**	1244	0	1255	0.884
TA3	15	15	(1206 1218)	**1206**	**0**	1222	1.327	1225	1.575
TA4	15	15	(1170 1175)	**1170**	**0**	1175	0.427	1191	1.795
TA5	15	15	(1210 1228)	**1215**	**0.41**	1229	1.57	1256	3.802
TA6	15	15	(1210 1239)	**1210**	**0**	1245	2.893	1247	3.058
TA7	15	15	(1223 1228)	**1223**	**0**	1228	0.409	1244	1.717
TA8	15	15	(1187 1217)	**1187**	**0**	1220	2.78	1222	2.949
TA9	15	15	(1247 1274)	**1297**	**0.4**	1291	3.528	1291	3.528
TA10	15	15	1241	**1241**	**0**	1250	0.725	1266	2.015
TA11	20	15	(1321 1364)	**1357**	**2.73**	1371	3.785	1402	6.132
TA12	20	15	(1321 1367)	**1367**	**3.48**	1379	4.391	1416	7.192
TA13	20	15	(1271 1350)	**1369**	**7.71**	1362	7.16	1377	8.34
TA14	20	15	1345	**1345**	**0**	1345	0	1361	1.19
TA15	20	15	(1293 1342)	**1348**	**4.25**	1360	5.182	1383	6.961
TA16	20	15	(1300 1368)	**1351**	**3.92**	1370	5.385	1418	9.077
TA17	20	15	(1458 1464)	**1458**	**0**	1481	1.578	1519	4.184
TA18	20	15	(1369 1396)	**1412**	**3.14**	1426	4.164	1433	4.675
TA19	20	15	(1276 1341)	**1336**	**4.7**	1351	5.878	1376	7.837
TA20	20	15	(1316 1353)	**1347**	**2.36**	1366	3.799	1398	6.231
TA21	20	20	(1539 1647)	**1649**	**7.147**	1659	7.797	1692	9.942
TA22	20	20	(1511 1601)	**1627**	**7.67**	1623	7.412	1638	8.405
TA23	20	20	(1472 1558)	**1556**	**5.7**	1573	6.861	1594	8.288
TA24	20	20	(1602 1651)	**1624**	**1.37**	1659	3.558	1714	6.991
TA25	20	20	(1504 1598)	**1580**	**5.05**	1606	6.782	1631	8.444
TA26	20	20	(1539 1655)	**1672**	**8.64**	1666	8.252	1698	10.331
TA27	20	20	(1616 1689)	**1688**	**4.45**	1697	5.012	1722	6.559
TA28	20	20	(1591 1615)	**1602**	**0.69**	1622	1.948	1653	3.897
TA29	20	20	(1514 1625)	**1583**	**4.55**	1635	7.992	1639	8.256
TA30	20	20	(1473 1596)	**1573**	**6.78**	1614	9.572	1621	10.048
TA31	30	15	(1764 1766)	**1764**	**0**	1771	0.397	1809	2.551
TA32	30	15	(1774 1803)	**1824**	**2.81**	1840	3.72	1840	3.72
TA33	30	15	(1778 1796)	**1829**	**2.87**	1833	3.093	1844	3.712
TA34	30	15	(1828 1832)	**1841**	**0.71**	1846	0.985	1898	3.829
TA35	30	15	2007	**2009**	**0.09**	2007	0	2010	0.149
TA36	30	15	(1819 1823)	**1825**	**0.32**	1825	0.33	1874	3.024
TA37	30	15	(1771 1784)	**1796**	**1.41**	1813	2.372	1846	4.235
TA38	30	15	(1673 1681)	**1699**	**1.49**	1697	1.435	1762	5.32
TA39	30	15	(1795 1798)	**1803**	**0.44**	1815	1.114	1822	1.504
TA40	30	15	(1631 1686)	**1684**	**3.25**	1725	5.763	1749	7.235
TA41	30	20	(1859 2023)	**2019**	**8.66**	2045	10.005	2106	13.287
TA42	30	20	(1867 1961)	**1956**	**4.76**	1979	5.999	2018	8.088
TA43	30	20	(1809 1879)	**1902**	**5.14**	1898	4.92	1946	7.573
TA44	30	20	(1927 2003)	**1987**	**3.11**	2036	5.656	2069	7.369
TA45	30	20	(1997 2005)	**2011**	**0.7**	2021	1.202	2049	2.604
TA46	30	20	(1940 2033)	**1997**	**2.94**	2047	5.515	2115	9.021
TA47	30	20	(1789 1920)	**1906**	**6.54**	1938	8.329	1973	10.285
TA48	30	20	(1912 1973)	**1982**	**3.66**	1996	4.393	2080	8.787
TA49	30	20	(1915 1991)	**1993**	**4.07**	2013	5.117	2046	6.841
TA50	30	20	(1807 1951)	**1975**	**9.29**	1975	9.297	2009	11.179
TA51	50	15	2760	**2760**	**0**	2760	0	2760	0
TA52	50	15	2756	**2756**	**0**	2756	0	2756	0
TA53	50	15	2717	**2717**	**0**	2717	0	2717	0

Table 12 (continued)

Problem	n	m	Opt (LB UB)	**AIS**	**RE_{AIS}**	TSSB	RE_{TSSB}	SB-GLS1	$RE_{SB\text{-}GLS1}$
TA54	50	15	2839	**2839**	**0**	2839	0	2839	0
TA55	50	15	2679	**2681**	**0.07**	2684	0.187	2683	0.149
TA56	50	15	2781	**2781**	**0**	2781	0	2781	0
TA57	50	15	2943	**2943**	**0**	2943	0	2943	0
TA58	50	15	2885	**2885**	**0**	2885	0	2885	0
TA59	50	15	2655	**2655**	**0**	2655	0	2657	0.075
TA60	50	15	2723	**2723**	**0**	2723	0	2723	0
TA61	50	20	2868	**2868**	**0**	2868	0	2891	0.802
TA62	50	20	(2869 2895)	**2895**	**0.9**	2942	2.544	2962	3.242
TA63	50	20	2755	**2755**	**0**	2755	0	2796	1.488
TA64	50	20	2702	**2702**	**0**	2702	0	2726	0.888
TA65	50	20	2725	**2725**	**0**	2725	0	2751	0.954
TA66	50	20	2845	**2845**	**0**	2845	0	2845	0
TA67	50	20	(2825 2826)	**2842**	**0.6**	2865	1.416	2841	0.566
TA68	50	20	2784	**2784**	**0**	2784	0	2785	0.036
TA69	50	20	3071	**3075**	**0.13**	3071	0	3071	0
TA70	50	20	2995	**2995**	**0**	2995	0	3004	0.301
TA71	100	20	5464	**5464**	**0**	5464	0	5464	0
TA72	100	20	5181	**5181**	**0**	5181	0	5181	0
TA73	100	20	5568	**5568**	**0**	5568	0	5568	0
TA74	100	20	5339	**5339**	**0**	5339	0	5339	0
TA75	100	20	5392	**5396**	**0.07**	5392	0	5392	0
TA76	100	20	5342	**5344**	**0.037**	5342	0	5342	0
TA77	100	20	5436	**5436**	**0**	5436	0	5436	0
TA78	100	20	5394	**5394**	**0**	5394	0	5394	0
TA79	100	20	5358	**5358**	**0**	5358	0	5358	0
TA80	100	20	5183	**5183**	**0**	5183	0	5183	0
TOTAL					**149.214**		204.8		294.4
Mean relative error (MRE)					**1.865**		2.56		3.68

iteration. Steps 3.2.1. to 3.2.5. constitute an iteration. This is repeated for the desired number of iterations.

3.4 Final results after 1500 iterations

The final job sequences and makespan obtained after reaching the lower bound value are given in Table 8. The computational time for TA 03 problem is 94 CPU s.

4 Results and discussion

The proposed AIS algorithm has been tested for 130 problem instances of various sizes collected in the following classes:

Class (i):
Five instances denoted as (ORB1-ORB5) due to Applegate and Cook [27] and five instances denoted as (ABZ5-ABZ9) due to Adams et al. [28].

Class (ii):
Forty instances of eight different sizes (n×m = 10×5; 15×5; 20×5; 10×10; 15×10;20×10; 30×10; 15×15) denoted as (LA01-LA40) due to Lawrence [29].

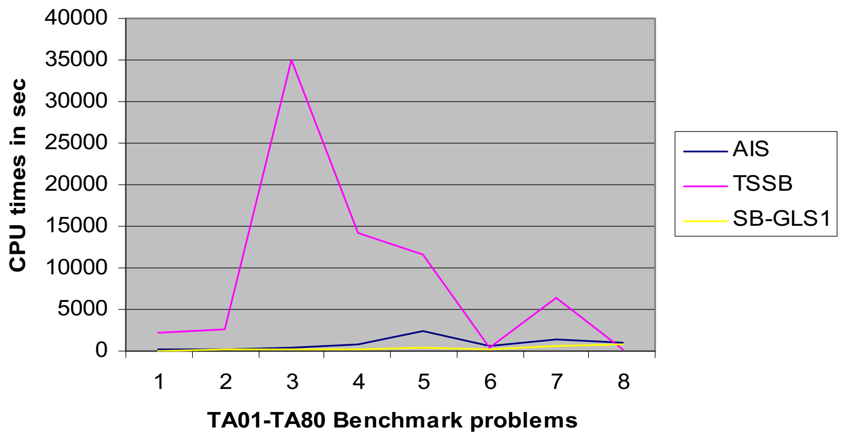
Class (iii):
Eighty instances of eight different size (n×m = 15×15; 20×15; 20×20; 30×15; 30×20; 50×15; 50×20; 100×20) denoted by Taillard (TA1-TA80) [30].

The relative error RE (%) was calculated for all problem instances, as a percentage by which the solution obtained is above the optimum value (Opt) if it is known or the best-known lower bound (LB) [19].

$$RE(\%) = 100 \times (UB - LB)/LB.$$

In Table 9, the solutions for class (i) problems obtained from AIS are compared with TSSB procedure and the mean relative error of AIS (0.305%) is found to be lower than the previously obtained results (1.378%) from TSSB procedure. AIS gives optimum bound value for 5 out of 10 problems

Fig. 7 The graph shows that the results obtained in AIS algorithm compared with TSSB and SB-GLS1 Procedure for problems TA-01 to TA-80



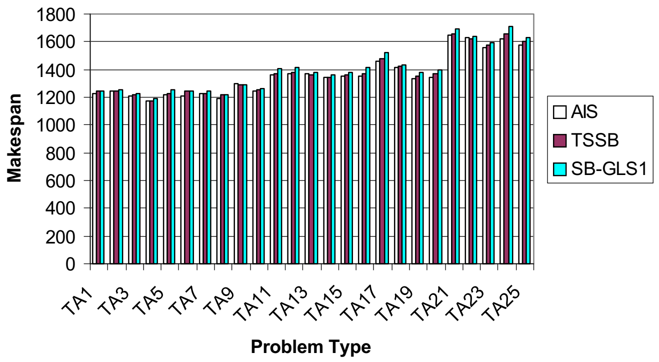


Fig. 8 The graph shows that the results obtained in AIS algorithm compared with TSSB and SB-GLS1 procedure for problems TA-1 to TA-26

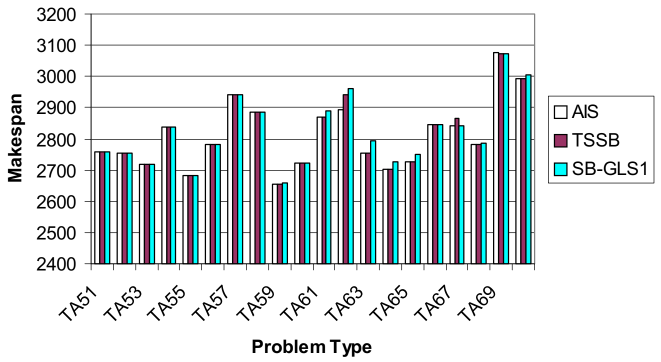


Fig. 10 Graph shows that the results obtained in AIS algorithm compared with TSSB and SB-GLS1 procedure for problems TA-51 to TA-70

where TSSB produces optimum value for 3 out of 10 problems. The graphical representation in Figs. 3 and 4 shows the comparison of class (i) problem results and computational time obtained from AIS with TSSB procedure [19].

In Table 10, the solutions for class (ii) problems obtained from AIS are compared with TSSB procedure and the mean relative error is found in AIS (0.048%) which is lower than the previously obtained value of 0.14% by TSSB procedure. AIS gives optimum bound value for 36 out of 40 problems where as TSSB produces optimum value for 33 out of 40 problems. The graphical representation in Figs. 5 and 6 shows the comparison of class (ii) problem results and average computational time obtained from AIS with TSSB procedure.

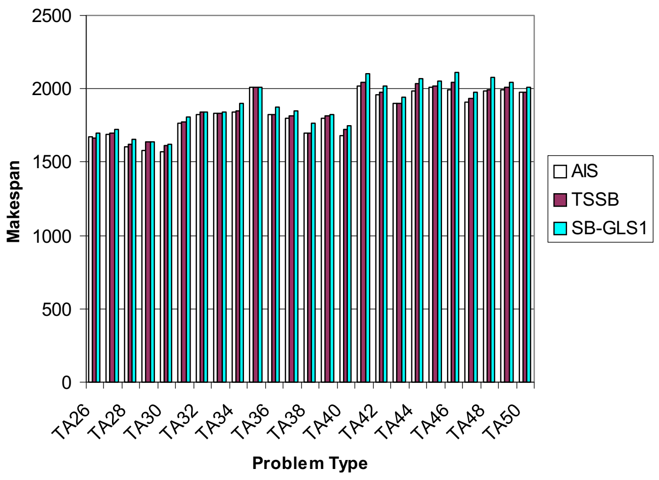


Fig. 9 The Graph shows that the results obtained in AIS algorithm compared with TSSB and SB-GLS1 procedure for problems TA-27 to TA-50

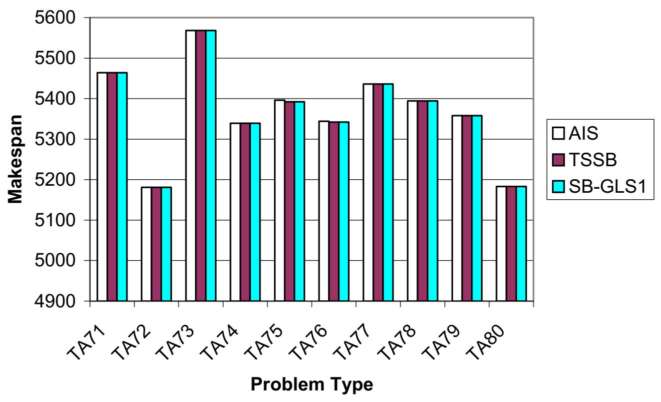


Fig. 11 Graph shows that the results obtained in AIS algorithm compared with TSSB and SB-GLS1 procedure for problems TA-71 to TA-80

Table 13 Comparison of mean relative error and computing time of AIS with TSSB and SB-GLS1 of Balas and Vazacopoulos [8]

Problem	n	m	AIS		TSSB		SB-GLS1	
			MRE	Tav	MRE	Tav	MRE	Tav
TA 01–10	15	15	0.08	118	1.45	2175	2.24	57
TA 11–20	20	15	3.23	232	4.13	2526	6.18	113
TA 21–30	20	20	5.21	495	6.52	34910	8.12	165
TA 31–40	30	15	1.34	835	1.92	14133	3.53	175
TA 41–50	30	20	4.89	2331	6.04	11512	8.50	421
TA 51–60	50	15	0.007	665	0.02	421	0.02	152
TA 61–70	50	20	0.163	1315	0.39	6342	0.83	590
TA 71–80	100	20	0.011	1019	0.00	231	0.00	851
			1.865		2.56		3.68	

In Table 12, the solutions of the results obtained in AIS are compared with TSSB, SB-GLS1 procedure. The mean relative error for class (iii) problem instances obtained by AIS is 1.865% which is lower than the previously obtained results of 2.56% from TSSB procedure and 3.68% from SB-GLS1 procedure. AIS gives optimum bound value for 55 out of 80 problems where as TSSB produces optimum value for 32 out of 80 problems and SB-GLS1 generates optimum value of 20 out of 80 problems. The graphical representation in Figs. 7, 8, 9, 10, 11 shows the comparison of class (iii) problem results and average computational time obtained from AIS with TSSB procedure and SB-GLS1 procedure. In Table 11 and 13 shows the mean relative error and average computational time obtained for class (ii) and class (iii) problems using of AIS, TSSB and SB-GLS1 procedure.

5 Conclusion

In this paper, the proposed AIS approach has been used for solving job shop scheduling problems with the objective of makespan minimization. The algorithm uses simple but effective techniques for calculating cloning process, applying mutations, and a receptor editing procedure. The algorithm has been tested on 130 benchmark problem instances. The percentage deviations from lower bounds were calculated. The findings were compared with the Tabu search shifting bottleneck procedure (TSSB) and SB-GLS1 procedure that tested the same problems. The proposed AIS algorithm is competent and proves to be a good problem-solving technique for job shop scheduling.

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. Math Oper Res 1:117–129
3. Erschler JF, Roubellat JP, Vernhes (1976) Finding some essential characteristics of the feasible solutions for a scheduling problem. Oper Res 24:774–783
4. French S (1982) Sequencing and scheduling: An introduction to the mathematics of the job shop. Wiley, New York
5. Carlier J, Pison E (1989) An algorithm for solving the job shop problem. Manage Sci 35:164–176
6. Bruker P et al (1994) A branch and bound algorithm for job shop problem. Discrete Appl Math 49:107–127
7. Blazewickz J (1996) The job shop scheduling problem: Conventional and new solutions techniques. Eur J Oper Res pp 931–30
8. Pezzella F, Merelli E (2000) A tabu search method guided by shifting bottleneck for the job shop scheduling problem. Eur J Oper Res 120:297–310
9. Balas E, Vazacopoulos A (1994) Guided local search with shifting bottleneck for job shop scheduling. Tech Rep Management Science Research Report MSRR-609, GSIA Carnegie Mellon University, Pittsburgh
10. Forrest S, Perelson A, Allen L, Cherukuri R (1994) Self-nonself discrimination in a computer. Proc IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, Los Alamitos, CA, pp 202–212
11. Forrest S, Hoffmeyr SA (2000) Engineering an immune system. Graft 4(5):5–9
12. Dasgupta D, Forrest S (1996) Novelty detection in time series data using ideas from immunology. Proc ISCA'96, Reno, Nevada, June 1996, pp 19–21
13. Dasgupta D, Forrest S (1999) Artificial immune systems in industrial applications. Proc Second International Conference on Intelligent Processing and Manufacturing Materials (IPMM), Honolulu, July 1999, pp10–15
14. De Castro LN, Von Zuben FJ (2000) The clonal selection algorithm with engineering applications. Proc Workshop on GECCO 2000, Las Vegas, 8-12 July 2000, pp 36–37
15. Trojanowski K, Wierzchon ST (2002) Searching for memory in artificial immune system. Proc 11th International Symposium on Intelligent Information Systems, June 2002, pp 3–6
16. Nasaroui O, Dasgupta D, Gonzales F (2002) The promise and challenges of artificial immune system based web-usage mining: preliminary results. Proc Workshop on Web Analytics at Second SIAM International Conference on Data Mining (SDM), Arlington, VA, April 2002, pp 11–13
17. Timmis J, Neal MJ (2000) A resource limited artificial immune system for data analysis. Research and Development in Intelligent Systems XVII, Proceedings of the ES2000, Cambridge, UK, pp 19–32
18. Yang S, Wang D (2001) A new adaptive neural network and heuristics hybrid approach for job shop scheduling. Comput Oper Res 28:955–971

19. Engin O, Doyen A (2004) A new approach to solve hybrid flow shop scheduling problems by artificial immune system. Future Generation Comput Syst 20:1083–1095
20. Costa AM, Vargas PA, Von Zuben FJ, França PM (2002) Makespan minimization on parallel processors: an immune based approach. Proc Special Sessions on Artificial Immune Systems, IEEE World Congress on Computational Intelligence, Honolulu, HI, pp 115–123
21. Zheng H, Zhang J, Nahavandi S (2004) Learning to detect texture objects by artificial immune approaches. Future Generation Comput Systems 20:1197–1208
22. Huang S (1999) Enhancement of thermal unit commitment using immune algorithms based optimization approaches. Electr Power Energy Syst 21:245–252
23. Zheng H, Zhang J, Nahavandi S (2004) Learning to detect texture objects by artificial immune approaches. Future Generation Comput Syst 20:1197–1208
24. Attux RRF, et al (2003) A paradigm for blind IIR equalization using the constant modules criterion and an artificial immune network. IEEE XIII Workshop on Neural Networks for Signal Processing, pp 839–849
25. May P, Mander K, Timmis J (2003) Mutation Testing: An Artificial Immune System Approach. Computing Laboratory, University of Kent, Canterbury, Kent, CT2 7NF
26. De Castro LN, Von Zuben FJ (1999) Artificial immune systems, Part 1, basic theory and applications. Technical Report, TR-DCA 01/99
27. Applegate D, Cook W (1991) A computational study of the job shop scheduling problem. ORSA J Comput 3:149–156
28. Adams J, Balas E, Zawack D (1988) The shifting bottleneck procedure for job shop scheduling. Manage Sci 34:391–401
29. Lawrence S (1984) Supplement to resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques. Tech Rep, GSIA, Carnegie Mellon Univ
30. Taillard E (1993) Benchmarks for basic scheduling problems. Eur J Oper Res 64:278–285