International Journal of Advanced Research in Engineering and Technology (IJARET)
Volume 10, Issue 1, January- February 2019, pp. 258-267, Article ID: IJARET_10_01_026
Available online at http://www.iaeme.com/IJARET/issues.asp?JType=IJARET&VType=10&IType=01
ISSN Print: 0976-6480 and ISSN Online: 0976-6499
© **IAEME** Publication

A NEW METHOD FOR SOLVING DEADLOCK USING GENETIC ALGORITHMS

Nada Thanoon Ahmed and Narjis Mezaal Shati
Computer Science Department, Collage of Science,
Mustansiriyah University, Baghdad, Iraq

ABSTRACT

A deadlock is reached when multiple processes share the same resource and prevent each other for accessing the resource. Resource organization is considered the optimal solution to solve this problem. This paper presents a novel approach of steady state genetic algorithms hybrid with banker's algorithm. The sequence of operation starts with features extraction that is used for feeding the Genetic algorithm optimizer. The chromosome structure in this method is the one operation processes correlated with three types of genetic operators: The one-point crossover (1X), two-point crossover (2X), and Uniform crossover (UX). Observe a large number of optimal solutions that avoids the Dead Lock system as a safe state.

Key words: Steady State Genetic Algorithm, Deadlock, Banker’s Algorithm, optimization.

Cite this Article: Nada Thanoon Ahmed and Narjis Mezaal Shati, A New Method for Solving Deadlock Using Genetic Algorithms, *International Journal of Advanced Research in Engineering and Technology, 10*(1), 2019, pp. 258-267.
http://www.iaeme.com/IJARET/issues.asp?JType=IJARET&VType=10&IType=1

1. INTRODUCTION

Synchronization objects produce a block for further execution processes. This phenomenon observed in many fields such as computer systems and Operations Management systems [1]. The deadlock researches classified it into four major types; the first type is the prevention deadlock, the avoidance deadlock [2], the detection deadlock and finally the recovery deadlock [3, 4]. The deadlock algorithm named as the Banker’s algorithm is used in banking system to serve the cash availability to satisfy the customer needs. The system methodology can be described when an existing system declares the maximum number of resource requirements based on the resource types. If the required resources do not exceed the query request, the system will be in safe state. But when the user requests are applied based on a set of resources, the system must determine whether the resources allocation will leave in safe state or not. If the process will allocate the resources, the state is safe. But if the process must wait for the releasing of some other resources, the phenomenon of deadlock occurs [5, 6]. Many approaches used to prevent this phenomenon; Genetic Algorithm is the most efficient one. Its approved as an effective method for solving and optimizing complicated problems. One of the state-of-the-art developments to the Genetic Algorithm is the Steady State Genetic Algorithm (SSG). In this improvement, the Gen produces on of the offspring in each generation. The fitness value will be measured in each offspring. After choosing the group of individuals randomly from the population, the efficiency will be compared with offspring efficiency. If the offspring present better best-fit tan the chromosome best-fit in the population, then the offspring best-fit will be replaced based on replacement strategy. After this process, the updated population is moved automatically to the next generation, otherwise the movement of the population will occur to the next change without being changed [7, 8, 9].

2. RELATED STUDIES

In any system design, the strategy of preventing deadlocks constructed by organizing the resources to ensure that at least one of the necessary conditions for deadlock can never hold. Many studies presented in this area. In [10], the researchers presented the reinforcement learning scheduling algorithm. This method correlates with high level deadlock detection and is applied in job-shop discrete manufacturing system. The system was without buffering and the first detection approach proposed to the second level and third level deadlocks. By continuing, the high-level deadlock detection algorithm developed in the context of less buffer of the job-shop system using the reinforcement learning scheduling algorithm. In [1], the researchers presented an effective sub-optimal deadlock avoidance policy. This efficient algorithm called the heuristic-based parameterized Banker’s algorithm (H-pBA). The present method observes a superior result due to integration of the first buffer, first serve policy in the system. In [11,12] a new scheduling method presented by the researchers. The method combines a robust supervised control with heretic search. The research goal is to minimize the make span of part list. Based on the reachability of the system, the present method establishes new heuristic function correlated with two dispatching rules. The researchers develop a dynamic polynomial window search algorithm.

In [13] the researchers focus on the automated manufacturing system deadlock control with the failure of multiple resources. The methodology presents two steps. The first step is process of siphon without unreliable resources. The solution was by adding optimal deadlock. Then, the unreliable resources controlled by adding new control to ensure that it could be marked when the resources failed.

3. SYSTEM METHODOLOGY

3.1. Developing the dataset

In this, research the dataset developed by using previous operational information. The collected data is organized to express the general deadlock obstacles in the real operation. Therefore, it can be a satisfactory indicator for evaluating the present method. Table 1, illustrate the numerical input representation of the present research.

Table 1 Maximum resource allocation and burst time

	R_0	**R_1**	**R_2**	**R_3**	**R_4**	**R_5**	**R_6**	**R_7**	**R_8**	**R_9**	**Burst time**
P0	15	22	25	10	7	15	10	14	12	19	30
P1	18	7	20	7	9	8	8	13	16	20	20
P2	17	10	24	8	8	25	7	12	16	18	25
P3	14	15	23	8	10	20	15	11	16	20	40
P4	10	20	20	6	12	21	13	9	13	19	22
P5	7	19	16	11	10	17	11	8	14	25	50
P6	8	21	15	10	11	11	12	12	10	20	30
P7	15	23	7	9	12	15	10	8	11	7	35
P8	19	5	18	2	13	13	9	7	12	15	10
P9	9	7	15	5	5	12	10	6	9	14	19

A hybrid of steady state genetic algorithms with banker's algorithm was designed and developed based on two stages, the first stage is by extracting the features and the second stage by applying Steady State Genetic Algorithms.

Stage1: Feature Extraction

In the proposed method, different types of inputs are fed to the Genetic algorithm optimizer (SSGA) such as: MAX array, ALLOCATION array, NEED array, RESOURCE vector, FREE RESOURCE vector, BURST TIME vector.

MAX Array: Consist of number of rows indicate the processes and the columns indicate the system resources and the intersection of rows and columns produce content the maximum instances of each type resources needed by the process, as shown in table (2).

Table 2 MAX Array

	R0	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**
P0	15	22	25	10	7	15	10	14	12	19
P1	18	7	20	7	9	8	8	13	16	20
P2	17	10	24	8	8	25	7	12	16	18
P3	14	15	23	8	10	20	15	11	16	20
P4	10	20	20	6	12	21	13	9	13	19
P5	7	19	16	11	10	17	11	8	14	25
P6	8	21	15	10	11	11	12	12	10	20
P7	15	23	7	9	12	15	10	8	11	7
P8	19	5	18	2	13	13	9	7	12	15
P9	9	7	15	5	5	12	10	6	9	14

ALLOCATION Array: Also consist of number of rows indicate the processes and the columns indicate the system resources and the cell produced from the intersection of rows and columns content the number of instances of each type of resources assigned to each process, as illustrate in table (3).

Table 3 ALLOCATION Array

	R0	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**R9**
P0	0	1	2	2	1	3	1	1	2	1
P1	1	1	4	0	1	2	1	0	1	0
P2	1	1	2	1	0	4	0	2	2	1
P3	0	1	1	1	1	3	0	2	2	2
P4	2	2	4	1	1	0	2	1	2	3
P5	1	2	1	1	1	2	1	0	0	3
P6	4	1	3	0	1	4	0	2	1	0
P7	0	0	1	1	1	3	2	1	1	4
P8	0	0	2	0	0	0	1	0	0	1
P9	4	2	3	1	2	2	1	1	1	2

NEED Array: 2D-array consist of rows as processes and the columns as resources. Each cell in this array content the number of instances of each type of resources needed for each process to complete its task. Equation (1) used to calculate the needed resource for each process. Table (4) illustrate the need array.

$$NEED_{ij} = MAX_{ij} - ALLOCATION_{ij} \quad ,\dots\dots\dots \qquad \dots\dots..(1)$$

Where: $i: 0, \dots, n,$

$j: 0, \dots, m,$

n: Number of processes

m: Number of Resources

Table 4 NEED Array

	R0	R1	R2	R3	R4	R5	R6	R7	R8	R9
P0	15	21	23	8	6	12	9	13	10	18
P1	17	6	16	7	8	6	7	13	15	20
P2	16	9	22	7	8	21	7	10	14	17
P3	14	14	22	7	9	17	15	9	14	18
P4	8	18	16	5	11	21	11	8	11	16
P5	6	17	15	10	9	15	10	8	14	22
P6	4	20	12	10	10	7	12	10	9	20
P7	15	23	6	8	11	12	8	7	10	3
P8	19	5	16	2	13	13	9	6	12	15
P9	5	5	12	4	3	10	9	5	8	12

RESOURCE Vector: 1D-array consist number of instances of all resources in the system, as seen in table (5) that illustrate the total number of instances of resources type in system.

Table 5 RESOURCE Vector

R0	R1	R2	R3	R4	R5	R6	R7	R8	R9
31	39	35	35	26	35	36	33	32	30

FREE RESOURCE Vector: Indicate number of free instances of all resources in system. It computed from ALLOCATION array by taking the summation of all resources assigned from all processes and subtract from the total number of system resources, as calculated in equation (2). The total number of free instances of each type of resources type in system illustrated in table (6).

$$FREE_j = RESOURCE_j - \sum_{i=0}^{n-1} ALLOCATION_{i,j} \quad ,\ldots\ldots \qquad ..(2)$$

Where: j: Number of resources [0..m-1],

m: Total number of resources,

i:: Number of process

n:: Total number of processes

Table 6 FREE RESOURCE Vector

R0	R1	R2	R3	R4	R5	R6	R7	R8	R9
18	28	12	27	17	12	28	22	20	14

BURST TIME Vector: Content the total time needed for each process to implement, as shown in table (7).

Table 7 BURST TIME Vector

P0	P1	P2	P3	P4	P5	P6	P7	P8	P9
30	20	25	40	22	50	30	35	10	19

Stage 2: applying Steady State Genetic Algorithms (SSGAs))

In this work, a steady state genetic algorithm optimizer using banker's algorithm in fitness adopted for the deadlock problem.

Chromosome Structure

In this research, the integer number gene used. It consist of the processes from 0 to p_{n-1} , whereas the chromosome length is the number of processes in the system (i.e. each gene is a process), as illustrated in figure (1). Whereas each chromosome indicates as a safe state or not safe in the system.

g_0	g_1	g_2	g_3	…	g_{n-1}
5	3	1	0	…	6

Figure 1 Chromosome Structure

Steps of new method using genetic Algorithm

In this work, the optimization Process is done as follows:

1. Initialize population randomly.
2. Evaluation (population) by using the following fitness equation:

$$f_{itness\ i} = \begin{cases} 0 & if\ the\ chromosome\ is\ safe\ state \\ -1 & otherwize \end{cases} \quad ,\ldots\ldots \qquad (3)$$

Where: i: Number of chromosomes in population.

Each chromosome must be checked if it safe state or not in the system. Each process (gene) in the chromosome will be examined if $FREE_j < NEED_{ij}$ then there is no enough resource so the fitness set to (-1) and stop checking the genes of this chromosome, and evaluate another chromosome. Otherwise, a NEWFREE array will be calculated for each process (genes in chromosome) according equation (4).

$$NEWFREE_j = FREE_j + ALLOCATION_{gi,j} \quad ,\ldots\ldots.. \qquad (4)$$

Where: gi : process number

j: resource number

3. Set the fitness of this chromosome to (0) and calculate a new burst time of this chromosome, as in equation (5):

For each chromosome:

$$Average\ Waiting\ Time = \frac{\left(\sum_{i=0}^{n-2} \ \sum_{k=0}^{i} \ burst\ time\,[g_i]\right)}{n} \quad ,\ldots \ (5)$$

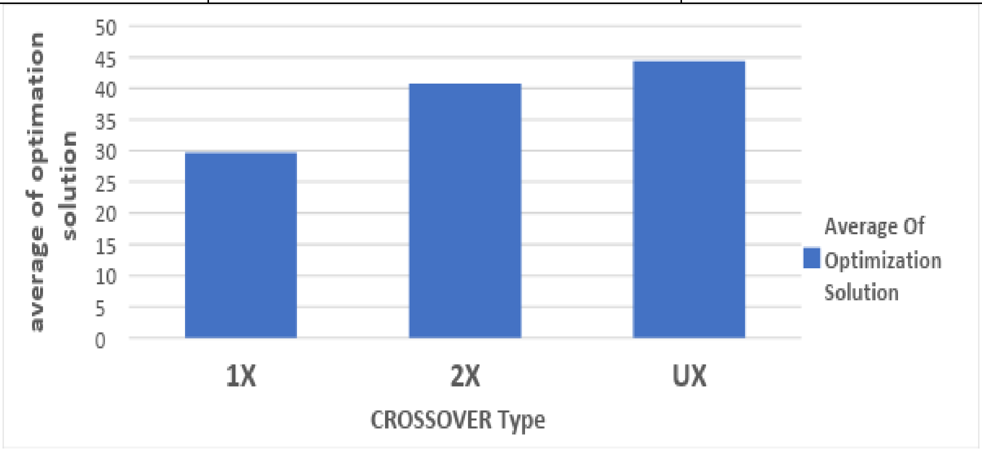
4. Repeat step (2) for all chromosomes in the population.
5. Selection: The Binary Tournament selection method will be used in this work,
6. Crossover: all crossover types used in this research work (one-point crossover (1X), two-point crossover (2X), Uniform crossover (UX)).
7. Mutation: one mutation at random position will be used in this work
8. Replacement: Binary Tournament Replacement will be used.
9. Stopping point: After running the genetic algorithm, stopping when the following conditions occurs: (Access to the maximum number of generations, when the whole population becomes a safe state).

4. RESULT AND DESICCATION

The proposed method tested on datasets in table (1). The results of average 10 time runs for the proposed method and Binary Tournament selection and for all crossover types shown in table (8) and figure (2). The results show that the proposed method is able to find a large number of optimal solutions, which represent the safe state of the system, which avoid the system of Dead Lock case.

Table 8 The result of average 10 runs of the proposed method

Crossover type	Average Of Optimization Solution	No. Of Generation
1X	29.7	853
2X	40.8	950.9
UX	44.4	894.2



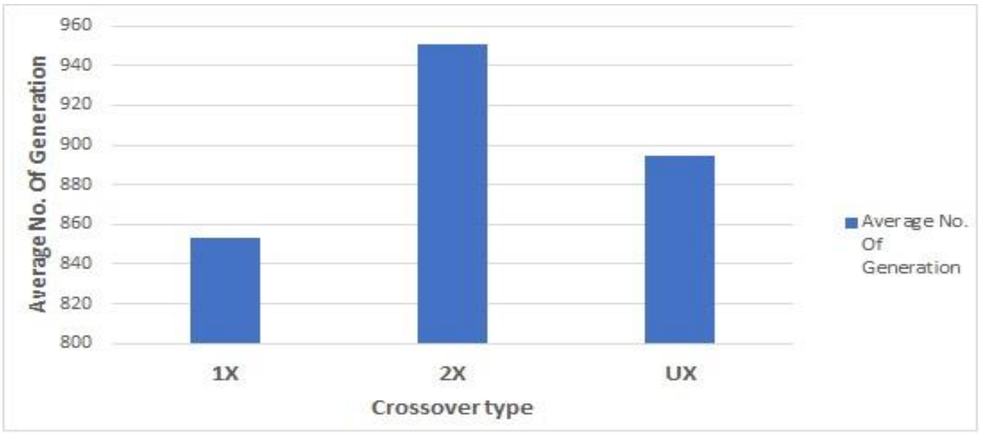


Figure 2 The bar chart of 10 runs average of the proposed method result

Table (8), illustrates that the test result of UX crossover type indicate that it is the best type to obtain almost all safe states of the system and a few generations compared to other crossover types.

Tables (9, 10, 11) and figures (3, 4, 5), shows the implementation of the proposed method using (One-point crossover (1X), Two-point crossover (2X), Uniform point crossover (UX)) respectively, each experiment repeated 10 runs.

Table 9 The effect of using one-point crossover (1X). (Repetition: 10 runs)

Run number	NO. Of Optimization Solution (safe state)	No. Of Generation
1	16	950
2	42	934
3	34	870
4	35	778
5	23	640
6	24	766
7	29	757
8	19	830
9	26	1050
10	49	955

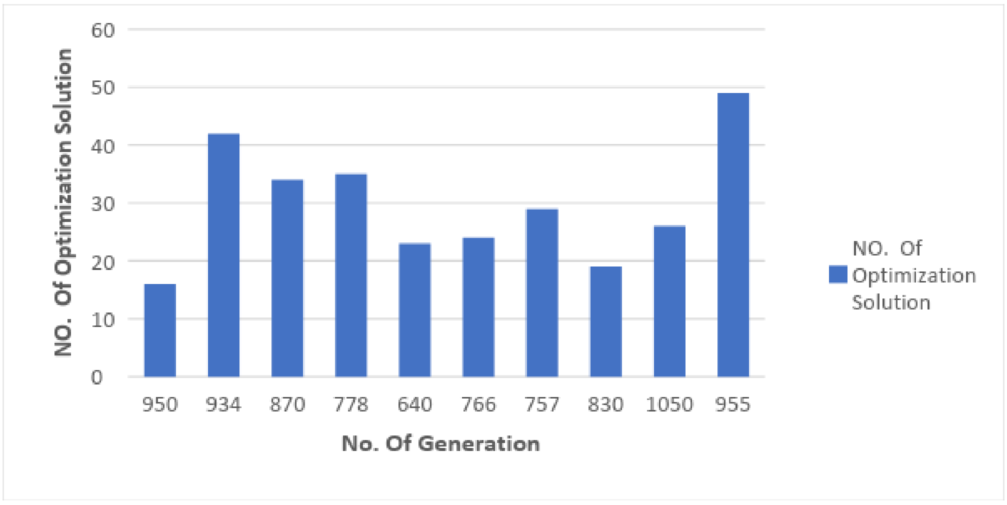


Figure 3 The bar chart of using one-point crossover (1X).

Table 10 The effect of using two-point crossover (2X). (Repetition: 10 runs)

Run number	NO. Of Optimization Solution (safe state)	No. Of Generation
1	57	1100
2	33	800
3	25	750
4	46	1013
5	20	893
6	32	921
7	59	1052
8	36	1005
9	64	1010
10	36	965

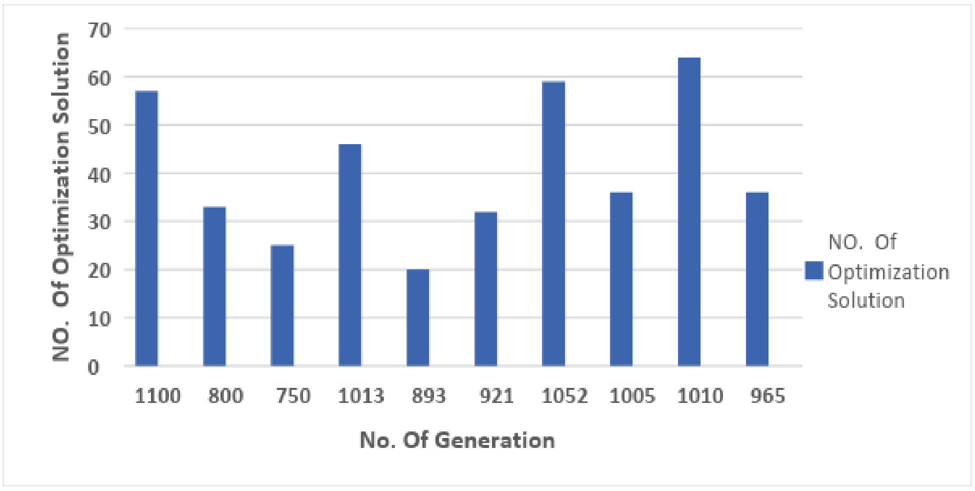


Figure 4 The bar chart of using two-point crossover (2X).

Table 11 The effect of using uniform point crossover (UX). (Repetition: 10 runs)

Run number	NO. Of Optimization Solution (safe state)	No. Of Generation
1	27	650
2	37	842
3	50	1004
4	38	724
5	26	708
6	55	1054
7	67	987
8	42	930
9	57	1070
10	45	973

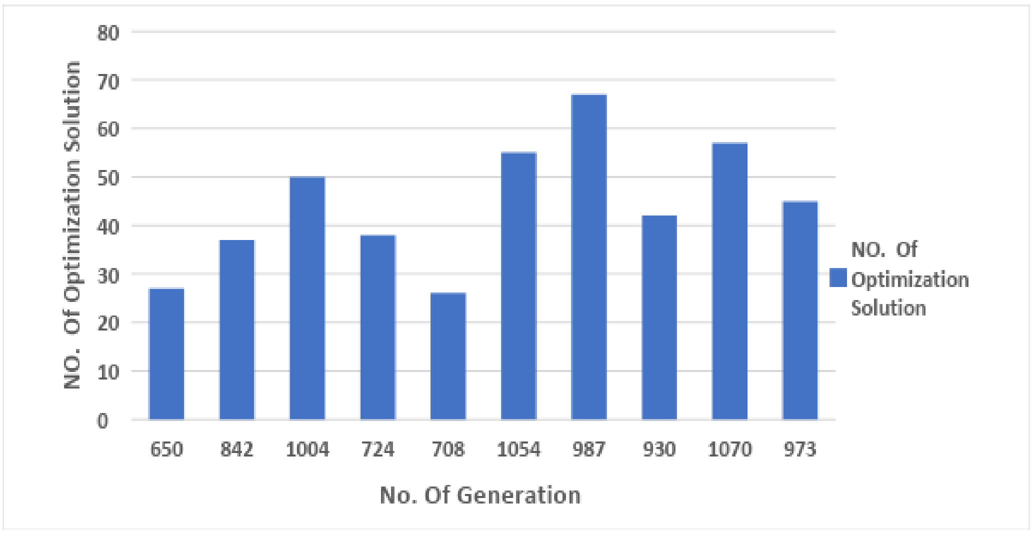


Figure 5 The bar chart of using uniform point crossover (UX).

From comparing the result illustrated in the previous tables and figures the following indication obtained:

- It is possible to obtain all the safe states of the system at the save time and then according to the waiting time average the best one chosen.
- The UX crossover type is the best in terms of the number of safe states
- The 1x crossover type is not good according to the number of safe states of the system obtained by using it.

5. CONCLUSIONS

The hybrid Technique that is proposed in this paper indicates, based upon, the experiment results that the proposed method is better than the banker's algorithm that when run obtains one safe state for the system. Whereas when using the proposed method a large number of safe states can obtained , the development of the banker's algorithm prevents the deadlock, so, the algorithm is efficient and can apply on different problems in different areas of life and we can indicate that the proposed method predicts all of the safe states that will prevent the failure of the deadlock.

ACKNOWLEDGEMENT

The authors would like to thank Mustansiriyah University (www.uomustansiriyah.edu.iq) Baghdad – Iraq for its support in the present work.

REFERENCES

[1] Choi, J. Y.; "Design and comparative performance analysis of a heuristic-based parameterised Banker's algorithm using the CRL scheduling problems"; International Journal of Production Research; Vol. 53, No. 9, 2605–2616, http://dx.doi.org/10.1080/00207543.2014.970710; 2015.

[2] Mo, T.; and Et al.; "Deadlock-free Production Scheduling with Dynamic Buffers in MES"; Smart Science; 2:4, 196-201, DOI: 10.1080/23080477.2014.11665626; 2014.

[3] Mota, V., E. S., L., A.; "Task Scheduling for Multiple Robots in an Industrial Environment"; Faculty of engineering; University of Porto; 2015.

[4] UMAR, U., A.; "Hybrid Multi objective Genetic Algorithm for Integrated Dynamic Scheduling and Routing of Jobs and Automated Guided Vehicles in Flexible Manufacturing Systems"; PhD. Dissertation; University Putra Malaysia; Malaysia; 2014.

[5] Bobanac, V.; " Routing and scheduling in multi-AGV systems based on dynamic banker algorithm"; Mediterranean Conference on Control and Automation - Conference Proceedings; MED'08, pages 1168–1173, doi:10.1109/MED.2008.4602057; 2008.

[6] Silberschatz, A.; and Et al.; "Operating System Concepts"; ninth Edition; New York: Wiley; ISBN 978-1118063330; USA; 2012.

[7] Shati, N. M.; "Hybrid Steady State Genetic Algorithm for Layout Problem and the Effect of using it on Some Types of Crossover"; Journal: Alustath, ISSN: 0552265X 25189263; Issue: 144 Pages: 473-496; Baghdad University; 2011.

[8] Goldberg, D. E.; "Genetic Algorithm in search, optimization, and Machine learning"; Addison-Wesley; 1989.

[9] Nada Thanoon Ahmed, Yasmin Makki Mohialden and Dina Riadh Abdulrazzaq; "A New Method for Self-Adaptation of Genetic Algorithms Operators"; International Journal of Civil Engineering and Technology; Volume 9, Issue 11, Pages 1279-1285; November 2018; http://www.iaeme.com/ijciet.

[10] Chen, M.; "Policy based reinforcement learning approach Of Job shop scheduling with high level deadlock detection"; MSc. Thesis; Iowa State University; Ames, Iowa; 2013.

[11] Luo, J. C.; and Et al.; "Scheduling of deadlock and failure-prone automated manufacturing systems via hybrid heuristic search"; International Journal of Production Research; http://dx.doi.org/10.1080/00207543.2017.1306132; 2017.

[12] Chen, M.; and Rabelo, L.; "Deadlock-Detection via Reinforcement Learning"; Ind Eng Manage; Vol.6: 215. doi:10.4172/2169-0316.1000215; 2017.

[13] Wu, Y.; and Et al.; "Robust deadlock control for automated manufacturing systems with a single type of unreliable resources"; Advances in Mechanical Engineering; Vol. 10(5) 1–14; DOI: 10.1177/1687814018772411; 2018.

[14] Prof. D. L. Bhombe and Mr. N. B. Bhawarkar, A Genetic Algorithm Applications for Multi-Objective Advanced Planning and Scheduling Problems - A Review, *International Journal of Electronics and Communication Engineering & Technology (IJECET),* Volume 4, Issue 6, November - December, 2013, pp. 93-106.

[15] Leena Jain and Amarbir Singh, A Genetic Algorithm Based Approach to Solve VLSI Floor planning Problem, *International Journal of Computer Engineering & Technology*, 9(6), 2018, pp.46–54.

[16] Leena Jain and Amarbir Singh, An Entropy Based Genetic Algorithm to Simultaneously Minimize Area and Wire length for VLSI Floor planning Problem International Journal of Advanced Research in Engineering and Technology, 9(6), 2018, pp 30–39.

[17] J. Srinu Naick and Dr K. Chandra sekar, Application of Genetic Algorithm and Neuro Fuzzy Control Techniques for Automatic Generation Control of Interconnected Power Systems and to Study the Development of a Hybrid Neuro Fuzzy Control Approach, *International Journal of Electrical Engineering and Technology (IJEET)*, Volume 4, Issue 4, July-August (2013), pp. 62-66.