IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 6, Ver. VIII (Nov – Dec. 2014), PP 18-25
www.iosrjournals.org

Neural Network for Solving Job-Shop Scheduling Problem

Miss. Rukhsana G. Sache
SVERI's College of Engg, Pandharpur Dist. Solapur, State: Maharashtra

Abstract: *The job shop scheduling is a very important scheduling problem, which is NP-complete in the strong sense and with well-known benchmark instances of relatively small size, which attest the practical difficulty in solving. Artificial neural network models have been successfully applied to solve such a job-shop scheduling problem (JSSP) known as a Nonpolynomial (NP-complete) constraint satisfaction problem. Our main contribution is an improvement of the algorithm proposed in the literature, which consists optimization of initial value of starting time. The main objective is to minimize the total weighted completion time of the jobs in the system that is the minimization of the makespan time by using the heuristic method. In this study, the heuristic method is used which gives a high quality approximate solution in reasonable time. The main advantage of using Hopfield Neural Network (HNN) is to improve the searching speed for getting an optimal or near optimal solution of a deterministic JSSP for reducing the makespan time. The simulation of the proposed method has been performed on various benchmarks. For two jobs and three machines (2/3/J/Cmax) dataset problem and any dataset problems, the simulation results shows the efficient with respect to the resolution speed, quality of the solution, and the reduction of the computation time which was not solved by Fanaiech et al. So, the simulation results have revealed that proposed heuristic algorithm can find high quality solutions to large sized instances very quickly.*

Keywords*: Artificial Neural Network, Constraints, Job shop scheduling, Heuristic, Hopfield network, Optimization methods.*

I. Introduction

The job-shop scheduling problem (JSSP) aims at a resource allocation problem with respect to allocation and sequence constraints. It is one of the most complicated and complex issues. Scheduling theory is concerned with the mathematical formulation and study of various scheduling models and development of associated solution methodologies. Some widely researched models are: the single machine model and its variants, parallel machine models, flow shop and job shop scheduling models. Of these, the deterministic job shop scheduling model has attracted the most attention for two key reasons. First, the generic formulation of the model makes it applicable to non-manufacturing scheduling domains. Second, the problem's sheer intractability has inspired researchers to develop a broad spectrum of strategies, ranging from simple heuristics to adaptive search strategies based on conceptual frameworks borrowed from biology, genetics and evolution. JSSP has a great impact on manufacturing systems nowadays. In the modeling and design of manufacturing systems, the quality of service (QoS) In order to evaluate the efficacy of the framework Zude [3] has developed a prototype Model effectively satisfy the various performance requirements for manufacturing systems , the design, and verification of distributed networks Hirsch et al , table-based scheduling method is one of the approach adapted to the distribute DSM is introduced by Handa et al. which are directly affected by applying JSSP new and efficient algorithms.

The most important standard optimization methods from the research in the literature is dedicated to the following methods such as heuristic methods Yahyaoui et al., genetic algorithms Morandin et al. [4], intelligent methods Zaied et al., Ferrolho et al[4], neural networks Foo et al., Satake et al. and Yang et al. [5], particle swarm optimization (PSO) Huynh et al., Liao et al., Electron et al. local search Ma et al., hybrid methods Moghaddam et al., and taboo search Yong et al. to mention but a few. References [6] provide excellent review papers on JSSP. Recently, the neural network approach for JSSP has been one of the hotspots in the production–scheduling field for a comprehensive classification and a review of production scheduling with neural network. First Foo et al. presented a novel approach using an original Hopfield Neural Network (HNN) for solving the JSSP. Then, they introduced integer linear programming networks as an extension of the original Hopfield network. Their objective is the minimization of the sum of all the starting times last operation (makespan) for each job. So far, HNN have been successfully applied to solve JSSP. All the works mentioned here used nonadaptive networks. Willems et al. [1]–[2], proposed a recent HNN structure used for solving a JSSP. They described the method and applied it on a simple example of five machines and five jobs.

In this paper, we propose a modification of the heuristic proposed by Williams et al. [1]–[2] Job-shop scheduling problem is one of the most complicated and typical issues. It belongs to the large class of nondeterministic polynomial time complete (NP-complete) problems. In general, an NP-complete problem is

one which for some input of size N takes a time proportional to at least 2N. Even the JSSP is not only NP-hard, it is one of the worst members in the class. An indication of this is given by the fact that one 10×10 (10 jobs/10 machines) problem formulated by Muthand Thompson remained unsolved for over 20 years. The traditional job-shop scheduling problem is defined as follows: there are n jobs to be processed through m machines in a described order under certain restrictive assumptions; each job must pass through each machine exactly one time. The job processing on a machine is called an operation and requires a duration known as the processing time. Minimizing the makespan Cmax will be considered as the optimization criterion in this paper, in other words, to minimize the floating time of jobs. The proposed modification consists in a suitable choice of the HNN initialization starting time of the JSSP. This optimal choice draws a fast training of the neural network and near-optimal solution. This new heuristic has been tested and compared with other existing methods on many benchmarks.

II. Neural Networks

Neural networks may be viewed as a collection of communicating simple processing elements. These elements are a functional abstraction of the neurons in the central nervous system. Here the elements are called units, rather than 'artificial neurons' to avoid the suggestion that any biological plausibility is claimed.

A unit is a simple processing element, connected to other units by its 'weighted' (dendritic) connections, sees Fig. 1.

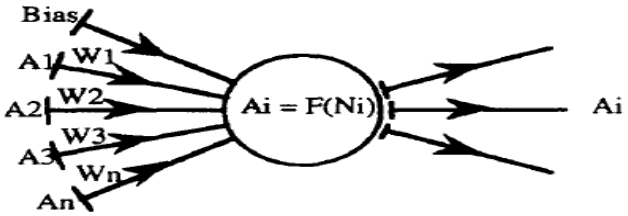

fig.1. unit [8]

A unit collects weighted (W1 to Wn) numerical information from other units (A1 to An). This information, sometimes increased with a bias, is summed resulting in the net input (Ni). The Ni is passed through an activation function F, resulting in the activation (Ai) of the unit.

The type of activation function implemented in a unit determines the functionality of the unit. After the summed input has been passed through the activation function, the activation (Ai) is collected by other units that are connected to this unit. The activation function may be (non) deterministic binary or (non)deterministic continuous. The activation function is selected according to the functionality required of the neural network.

Neural networks can perform two basic functions: they can be trained to remember some information [9], and they can be used to perform constraint satisfaction and optimisation tasks [10]-[12]. The job shop scheduling problem will be tackled by the latter type.

Hopfield and Tank [10] introduced a deterministic neural network model capable of solving constraint satisfaction and optimisation problems by translating the problem in a number of units with predefined fixed weighted connections. Examples of problems that have been solved in this way are the 'eight queens problem' and the 'travelling salesman problem' [10], crew scheduling [12] and others.

III. Job-Shop Scheduling

The job-shop contains universal resources combined with a great route flexibility. The jobs that arrive at a job shop consist of a predescribed, rigid sequence of operations (recipe). Which resource an operation has to be performed on, the allocation decision, is determined by the scheduler. The order in which the different operations of the available jobs have to be performed by the resources, the sequence decision, is also determined by the scheduler. The scheduler has to create a schedule that fulfils a specified performance criterion, resulting in the optimisation of the defined manufacturing system objectives. Different manufacturing systems with different objectives, or even different products, require different scheduling criteria. Several performance measures such as stock size, maximum throughput, due date reliability and mean lead time, are accepted as the main scheduling performance criteria. Several other criteria can be derived from these criteria, such as minimization of the makespan, minimization of the maximum lateness, or minimization of the summed lateness. Although the machine utilization degree remains widely accepted as a criterion, it appears to be only the resultant of an optimisation criterion. Which measures or derivatives are used as a performance criterion of the scheduler, depend on the manufacturer's intentions.

The search for an optimum schedule is bound by two types of constraints. The first constraint states that the compulsory operation sequence (recipe) is to be guaranteed; the second constraint states that not more

than one job can be processed on one resource at the same time. Scheduling can be viewed as optimisation, bound by sequence and resource constraints.

Job-shop scheduling belongs to the class of np complete problems. This means that the calculation effort required to find a solution to a job-shop scheduling problems grows exponentially or faster with the growth of the problem size. The class of np complete problems contains many other problems, like the scheduling of operators to machines, managers to departments, instructors to courses, etcetera. Job-shop scheduling has been extensively studied partly because it is an instance of the np complete class of problems.

Several notation systems have been presented for the representation of a specific job-shop scheduling problem, with its criterion. In this paper the notation system of Conway et al. [13] will be used. This is a convenient way of presenting a specific scheduling problem in which:

n = number of jobs

m = number of machines

A = operation pattern (e.g. J = Job-shop)

B = performance criterion (e.g. Cmax = minimization of the maximum completion time, equaling the minimization of the makespan).

A two-job, three-machine, job-shop scheduling problem with minimization of the makespan as performance criterion is represented as: 2/3/J/Cmax.

IV. Problem Formulation

Due to the difficulties encountered in solving job shop scheduling problems, several representational aids have been developed to facilitate schedule generation.

The constraints, a predefined operation sequence (sequence constraints) and the constraint that it is not allowed to operate more than one job on one resource the same time (resource constraints), can be translated to integer linear programming format in the following manner. The triplet ijk will be used to represent the operation j of job i on machine k. For each ijk a processing time t_{ijk} is known, and after the scheduling for each ijk a starting time is determined, denoted by S_{ijk}. The objective of jobs hop scheduling is to find a set of starting times (S_{ijk}) which comply with the constraints, minimizing the objective criterion.

To represent the constraints of a job-shop scheduling problem in an integer linear format, the sequence constraints have to be represented first. Suppose that the recipe of a job i states that operation j of job I requiring machine l is to be performed after operation (j-1) of job i requiring machine k is finished. Then, for a feasible schedule, it is necessary that $S_{ijl} \geq S_{i(j-1)k} - t_{i(j-1)k}$, which can be reformulated as $S_{ijl} - S_{i(j-1)k} - t_{i(j-1)k} \geq 0$ (1)

There are n(m-1) sequence equations for a n/m/J/B job-shop scheduling problem.

The general resource constraint is of a disjunctive type that can be represented by two equations. If job I precedes job p on machine k then operation ijk must be finished before job p can be processed on machine k. $S_{pjk} \geq S_{ijk} + t_{ijk}$, which can be reformulated as

$S_{pjk} - S_{ijk} - t_{ijk} \geq 0$ (2)

If, on the other hand, job p precedes job i on machine k then $S_{ijk} \geq S_{pjk} + t_{pjk}$, which can be reformulated as

$S_{ijk} - S_{pjk} - t_{pjk} \geq 0$ (3)

Only one of these two statements is valid (disjunctive statements), depending on the (partial) schedule generated. To accommodate these two constraints in the representation, an indicator variable, Y_{ipk}, is formulated.

This variable indicates which statement is valid, i.e. it states whether job i precedes job p on machine k (value 1) or not (value 0). Thus the resource constraints become:

$S_{pjk} - S_{ijk} + H(1 - Y_{ipk}) - t_{ijk} \geq 0$ (4)

$S_{ijk} - S_{pjk} + H * Y_{ipk} - t_{pjk} \geq 0$ (5)

with: $Y_{ipk} = 1$ if $S_{ijk} \leq S_{pjk}$

$Y_{ipk} = 0$ if $S_{ijk} > S_{pjk}$

The constant H should be large enough to ensure that one of the disjunctive statements holds while at the same time the other statement is eliminated due to H. To fulfill this requirement, H should be larger than the largest possible starting time difference between two operations. This value can be calculated by adding all processing times (worst possible schedule): $H > \sum_{i=1}^{n} \sum_{j=1}^{m} t_{ijk}$ (6)

There are nm(n-1) resource constraint equations for a job-shop scheduling problem, resulting in a total of n(mn-1) equations for a job-shop scheduling problem. These general integer linear equations representing the constraints will be used for the translation of specific job-shop scheduling problems to a format that can be mapped on a neural network.

The job-shop scheduling neural network should contain units that are capable of representing the starting times of the operations (S units), whether sequence constraints are violated (SC units), whether resource constraints are violated (RC units), and the value of the Yipk indicator variables (Y units).

4.1 Connections

First the connections to the SC units will be described. The constraint equation: $S_{ijb} - S_{i(j-1)a} - t_{i(j-1)a} \geq 0$ is represented by an SC unit. The SC unit collects the activation (representing the S_{ijk}) from the adequate S units of the bottom layer. To do so correctly, the S unit representing the starting time S_{ijb} should be connected with a weight of +1 and the S unit representing $S_{i(j-l)a}$ with a weight of -1. See Fig. 2. The bias of the SC unit consists of the negated $t_{i(j-1)a}$. The received input, together with the bias of the SC unit, indicates whether the represented constraint is violated by the suggested starting times. If the constraint is violated, the activation of the unit becomes greater than zero. This activation should be applied as a corrective signal for the S units; S_{ijb} should be increased and $S_{i(j-\ 1)a}$ should be decreased, resulting in a positive weighted (e.g. +0.1) feedback connection to S_{ijb} and a negative weighted (e.g. -0.1) feedback connection to $S_{i(i-1)a}$. See Fig. 2.

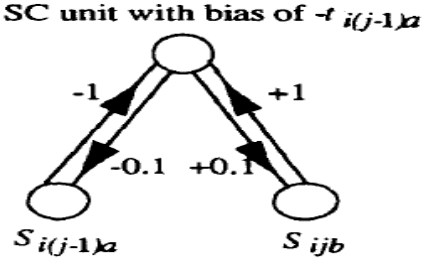

fig. 2. General structure for sequence constraints [8]

For the implementation of the resource constraints in a general unit type as described above, the value of H should be implemented in the bias of the RC unit.

This requires a reformulation of the H containing part of the constraint equation:

$H(1 - Y_{ipk}) = (-H * Y_{ipk}) + H,$

allowing the implementation of H as a bias and a weight value of the connection to the Y_{ipk} unit.

$$S_{pjk} - S_{ijk} + \left(-H * Y_{ipk}\right) + H - t_{ijk} \geq 0 \quad (7)$$

$$S_{ijk} - S_{pjk} + \left(+H * Y_{ipk}\right) - t_{pjk} \geq 0 \quad (8)$$

In this way the RC equation can be implemented in a general RC unit with the underlined part implemented as a bias, and H as a positive or negative weight value of the connection to Y_{ipk}.

The Y units should receive information from the S units such that S_{ijk} preceding S_{pjk} results in an activation of 1. Therefore the S_{ijk} representing unit should be connected with a negative (-1) weighted connection and the S_{pjk} representing unit with a positive (+1) weighted connection. See Fig. 3.

The RC units should receive information from the Y units as prescribed by the equations resulting in a connection weight of -H for the RC unit representing the first equation (7) and of +H for the RC unit representing the second (8) equation. See Fig. 3. Furthermore the RC units have to receive information from the S units. The RC unit representing the first equation should have a positive (+1) connection weight to the S_{pjk} representing S unit and a negative (-1) weighted connection to the S_{ijk} representing S unit. The RC unit representing the second equation should have a negative (-1) connection weight to the S_{pjk} representing S unit and a positive (+1) weighted connection to the Sijk representing S unit. This information, together with the underlined part of the equation as a bias of the respective RC units, suffices to determine whether the proposed starting times violate the represented constraint.

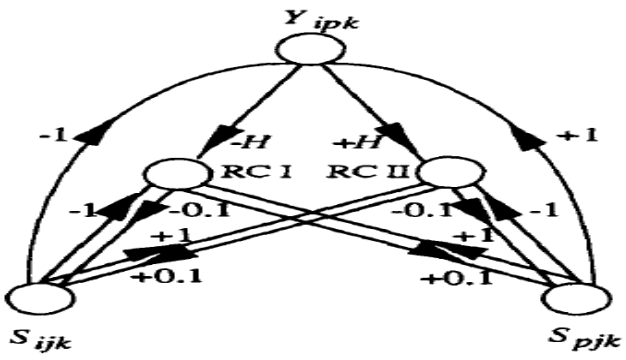

fig. 3. General structure for resource constraints [8]

A violated resource constraint should result in a modification of the starting times of the responsible S units. To achieve this, the S units collect the activation of the RC units through weighted connections to the RC units. The sign of the weights of these feedback connections cannot be unequivocally prescribed. If two operations are scheduled on one resource at the same time, it is not easily determined which operation should be advanced and which operation should be delayed. Here some rules of thumb like 'shortest processing time first' can be implemented by the correct setting of the weights. Small weights of these connections diminish the chance of getting stuck in a local minimum, at the cost of longer processing times.

V. Results

5.1 Implementation and Demonstration for 2/3/J/Cmax Dataset

This problems contains the nine main processing steps as Read input matrix (machine file & processing time file) , check input matrix is valid or not, calculate starting time (S_{ijk}), Apply Sequence constraints(equation (9)) to calculate Regular and Feedback matrix and Threshold value, Apply Resource constraints(equation (10) & (11)) to calculate Regular and Feedback matrix and Threshold value, Calculate Y Bias value , Check constraint satisfication, Calculate makespan time Cmax and last print the sequence .

For 2/3/J/Cmax scheduling problem is used with its machine allocation and operation times presented in table 1 and table 2.

Table 1 Machine assignments of 2/3/J/Cmax [8]

Machine Assignment		Operation		
		1	2	3
Jobs	1	1	2	3
	2	3	1	2

Table 2 Processing times of 2/3/J/Cmax [8]

Processing Times		Operation		
		1	2	3
Jobs	1	5	8	2
	2	7	3	9

After reading check these input matrices are valid or not, i.e. number of rows and columns in Machine assignment file is equal to number of rows and columns in processing time file.

The job-shop scheduling neural network should contain units that are capable of representing the starting times of the operations (S units), whether sequence constraints are violated (SC units), whether resource constraints are violated (RC units), and the value of the Y_{ipk} indicator variables (Y units).

Next step is to calculate the starting time (S units) of all jobs on all machines, therefore, the problem consists, firstly, of initializing the different starting times. For example, if the example of 2/3/J/Cmax is used, the corresponding starting times are: S_{111}, S_{122}, S_{133}, S_{213}, S_{221}, and S_{232}. Where S_{111} indicates starting time of 1 job of 1 operation on machine 1.

The objective of job-shop scheduling is to find a set of starting times S_{ijk} which comply with the constraints taking the minimization of the makespan as an optimization criterion by the mathematical problem formulation of the job-shop scheduling.

So, S_{ijk}= n*m

Where n= number of jobs

m=number of machines

For 2/3/J/Cmax example S_{ijk}= 2*3=6

So, the 6 sequences are generated and their starting time value in the form of matrix is shown below.

S matrix values

[1#1#1, 1#2#2, 1#3#3, 2#1#3, 2#2#1, 2#3#2] : [0.0, 5.0, 13.0, 0.0, 7.0, 10.0]

In above matrix 1#1#1, 1#2#2…… indicate the job, operation and machine.

Next step is to apply the Sequence constrains which mean that jobs must be processed on machines in a fixed sequence defined by process planning designers. Practically, the j^{th} operation of the job i supposed on machine l must be behind the $(j-1)^{th}$ operation of job i supposed on machine k. So for a feasible schedule, it is necessary that

$$S_{ijl} \geq S_{i(j-1)k} + t_{i(j-1)k}$$

This can be reformulated as

$$S_{ijl} - S_{i(j-1)k} - t_{i(j-1)k} \geq 0 \qquad (9)$$

There exist n (m − 1) sequence equations for an n/m/J/Cmax job-shop scheduling problem.
So, SC=n*(m-1) = 4 for 2/3/J/Cmax example sequence constraints of type

$$S_{ijl} \geq S_{i(j-1)k} + t_{i(j-1)k}$$

1) S122- S111- 5 ≥0
2) S133- S122 - 8 ≥0
3) S221- S213- 7 ≥0
4) S232 -S221- 3 ≥0.

So, after applying S constraints on 2/3/J/Cmax problem the output generated is shown in figure 4.

Regular SC matrix

-1.0	1.0	0.0	0.0	0.0	0.0
0.0	-1.0	1.0	0.0	0.0	0.0
0.0	0.0	0.0	-1.0	1.0	0.0
0.0	0.0	0.0	0.0	-1.0	1.0

Feedback SC matrix

-0.1	0.1	0.0	0.0	0.0	0.0
0.0	-0.1	0.1	0.0	0.0	0.0
0.0	0.0	0.0	-0.1	0.1	0.0
0.0	0.0	0.0	0.0	-0.1	0.1

fig. 4: SC Regular and Feedback matrix of 2/3/J/Cmax.

So, in SC Regular and Feedback matrix, in feedback matrix W_f= 0.1 is Small value of the weights of the feedback connections. In Fig 5 The first unit of the second layer (representing the first sequence equation) for instance, collects negated information (connection weight -1) from the unit representing S111 and positive information (weight +1) from the unit representing S122. Together with the bias of this unit (-5) the violation of this constraint can be determined. If, for instance, S111 = 1 and S122 = 2, a constraint violation should be signaled since the second operation starts before the first operation has ended. The net input of the first constraint unit, SC1 in Fig. 5, will be 2 - 1 - 5 = -4, resulting in an activation of 4, signaling a violation. This information has to be feedback to the S units to cause an appropriate change in starting times. For this reason, the S units collect the information from the SC units (see Fig. 5). The corresponding S units will receive this redirecting information resulting in an inhibition (+4 * -1) of the S111 unit and an excitation (+4 * 1) of the S122 unit, thus working towards an acceptable solution. If these feedback weights are set correctly, feasible solutions will be generated without requiting explicit initialization of the S units.

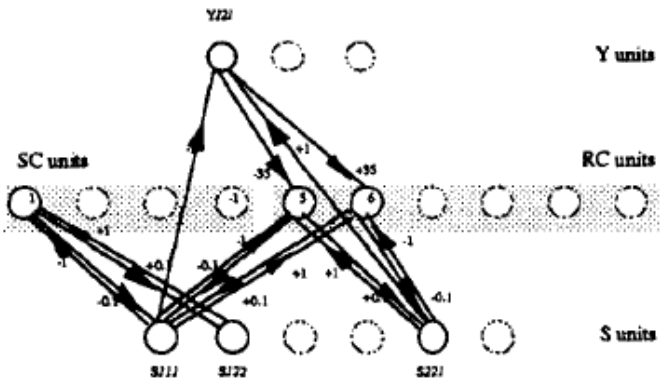

fig 5: 2/3/J/Cmax neural network [8].

Next step is to apply the Resource constrains which mean that at any one time, only one operation should be running on a machine. So, for the resource constraints the value of the constant H must be determined. This constant should have a value that is large enough to ensure that one of the disjunctive statements holds and the other is eliminated so:

$H > \sum_{i=1}^{n} \sum_{j=1}^{m} t_{ijk}$ H>34, H=35.

For the example problem there are nm (n-1) = 6 resource constraints of type

$$S_{pjk} - S_{ijk} + (-H * Y_{ipk}) + H - t_{ijk} \geq 0 \quad (10)$$
$$S_{ijk} - S_{pjk} + (+H * Y_{ipk}) - t_{pjk} \geq 0 \quad (11)$$

5) S221 -S111 + (-35*Y121) + 35 - 5 ≥0
6) S111 - S221 + (35* Y121) -3 ≥0
7) S232- S122 + (-35*Yl22) + 35 - 8 ≥0
8) S122 - S232 + (35* Y122) -9≥0
9) S213 - S133 + (-35*Y123) + 35 - 2≥0
10) S133 - S213 + (35* Y123) -7 ≥0.

So, after applying R constraints on 2/3/J/Cmax problem the output generated is shown in Fig 6.

Regular RC matrix

-1.0	0.0	0.0	0.0	1.0	0.0
1.0	0.0	0.0	0.0	-1.0	0.0
0.0	-1.0	0.0	0.0	0.0	1.0
0.0	1.0	0.0	0.0	0.0	-1.0
0.0	0.0	-1.0	1.0	0.0	0.0
0.0	0.0	1.0	-1.0	0.0	0.0

Feedback RC matrix

-0.1	0.0	0.0	0.0	0.1	0.0
0.1	0.0	0.0	0.0	-0.1	0.0
0.0	-0.1	0.0	0.0	0.0	0.1
0.0	0.1	0.0	0.0	0.0	-0.1
0.0	0.0	-0.1	0.1	0.0	0.0
0.0	0.0	0.1	-0.1	0.0	0.0

fig 6: RC Regular and Feedback matrix of 2/3/J/Cmax.

The RC units collect information from the adequate S units and the Y units according to the resource equations.

The thresholds of the S units can be determined in a problem-specific manner. For instance the threshold of the S unit representing the third operation of job 1 (S133) can be set at 13 (t111 + t122).

SC Threshold matrix values for 2/3/J/Cmax is

[-5.0, -8.0, -7.0, -3.0]

RC Threshold matrix value for 2/3/J/Cmax is

[-5.0 -3.0 -8.0 -9.0 -2.0 -7.0]

Next step is to calculate Y Bias i.e. Y_{ipk} value which is an indicator "zero-one" variable Y_{ipk} that indicates which job proceeds on machine k, $Y_{ipk} = 1$ if $S_{ijk} \leq S_{plk}$ and $Y_{ipk} = 0$ if $S_{ijk} > S_{plk}$ then the resource constraints become:

So, Y_{ipk} = mn (n-1)/2=3 for 2 job 3 machine problem.

For 2/3/J/Cmax problem the Value for Ybias =

1#2#1 = 1.0
1#2#2 = 1.0
1#2#3 = 0.0

Next step is after checking the SC and RC constraints, if violation is happened means constraints are not satisfy then update the starting time and again generate or check the SC and RC constraints.
And the last step is to calculate the makespan time C_{max} and print the sequence of jobs on machines.
So, for 2/3/J/Cmax problem value of Cmax : 22.11 and the Sequence of job in machine is shown below

Machine 1

1#1#1(0.0[5]) 2#2#1(7.0[3])

Machine 2

1#2#2(5.04[8]) 2#3#2(13.11[9])

Machine 3

2#1#3(0.0[7]) 1#3#3(13.11[2])

VI. Conclusion

Job shop scheduling problems fall into the class of NP-complete problem, they are among the most difficult to formulate and solve. But proposed methods, gives solution for 2 jobs and 3 machines (2/3/J/Cmax) and any datasets. And proposed method gives better result (minimum Cmax) as compare to Fanaiech et al.

A new proposed heuristic initialization procedure has several advantages in terms of reducing the number of cycles, difference between initial value of S unit (starting time) and final value of S unit. Hence it concludes initialization procedure is so important for better results.

In this simulation, 2/3/J/Cmax dataset problems are solved by using Hopfield Neural Network (HNN). New proposed heuristic method is used for calculating the initialization of the starting time of all the jobs and minimizing the makespan time. For these 2/3/J/Cmax dataset problems, the minimum makespan time generated is 22.11 respectively.

Reference

[1] T.M Willems and J.E Rod, " Neural nets for job shop scheduling will they do the job?" Proc. IFAC 12th World Conf., 1993, vol. 3, pp. 53–56.

[2] T.M.Willems and J. E. Roda, "Neural networks for job-shop scheduling," Control Eng. Pract., vol. 2, no. 1, pp. 31–39, Feb. 1994.

[3]] J. Käschel, T. Teich, G. Köbernik, and B. Meier, "Algorithms for the jobshop scheduling problem: A comparison of different methods," in Proc. Eur. Symp. Intell. Techn., Chania, Greece, Jun. 1999.

[4] A. Ferrolho and M. Crisostomo, "Intelligent control and integration software for flexible manufacturing cells," IEEE Trans. Ind. Electron., vol. 3, no. 1, pp. 3–11, Feb. 2007.

[5] S. Yang and D. Wang, "Constraint satisfaction adaptive neural network and heuristics combined approaches for generalized job-shop scheduling," IEEE Trans. Neural Netw., vol. 11, no. 2, pp. 474–485, Mar. 2000.

[6] D. E. Akyol and G. M. Bayhan, "A review on evolution of production scheduling with neural networks," Comput. Ind. Eng., vol. 53, no. 1, pp. 95–122, Aug. 2007.

[7] Amel Yahyaoui, Nader Fnaiech, and Farhat Fnaiech, "A Suitable Initialization Procedure for Speeding a Neural Network Job-Shop Scheduling", IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 58, NO. 3, MARCH 2011

[8] Wiilems M. and Rooda J. E., "Neural network for job shop scheduling," Control engineering practice, 2, pp. 31-39, 1994.

[9] Rumelhart , D.E., and McClelland, J.L. (1986). Parallel Distributed Processing: Explorations in the micro structure of cognition, Vol 1: Foundations, MIT Press.

[10] Hopfield, J.J. and Tank, D.W. (1985). "Neural" Computation of Decisions in Optimization Problems, Biol. Cybern., 52, 141-152.

[11] Tank, D.W., and Hopfield, LJ. (1986). Simple "Neural" optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit, IEEE Trans. Circuits and Systems, 33, 533-541.

[12] Poliac, M.O., Lee, E.B., Slagle, J.R. and Wick, M.R. (1987). A Crew Scheduling Problem, Proc. IEEE First Int. Conf. Neural Networks, pp. 779-786.

[13] Conway. R.W., Maxwell, W.L. and Miller, L.W. (1967). Theory of Scheduling, Reading, Addison- Wesley Publishing Company.