Asian Journal of Scientific Research

ISSN 1992-1454

science alert
http://www.scialert.net

ANSInet
an open access publisher
http://ansinet.com

Asian Journal of Scientific Research 1 (2): 146-152, 2008
ISSN 1992-1454

Quality Function Deployment in Agile Parallel Machine Scheduling Through Neural Network Technique

[1]S. Venkatachalam, [1]C. Arumugam, [2]K. Raja and [1]V. Selladurai
[1]Department of Mechanical Engineering, Coimbatore Institute of Technology, Coimbatore-641014, India
[2]Department of Mechanical Engineering, Kumaraguru College of Technology, Coimbatore-641006, India

Abstract: Any manufacturing system has to attain key performance measures for its successful operation. Quality Function Deployment (QFD) is to convert the customer requirements into quality characteristics and develop a schedule for the jobs by systematically deploying the relationships between the due date and the completion time by adopting the just in time concept. Non-traditioual optimization technique such as Neural Network (NN) technique provides a complete solution methodology for solving the shop floor scheduling problems. The problem considered in this study is to schedule different number of jobs on parallel machines with the objective of reducing the multiple objectives such as the earliness, the tardiness and the completion time of the jobs. All the objectives have been assigned with weights so that the priority of the objectives could be varied. It has been found that the proposed method simultaneously reduces all the performance measures considerably, thereby outperforming the existing heuristics.

Key words: Parallel machine scheduling, quality function deployment, neural network, earliness, tardiness

INTRODUCTION

This research addresses the agile parallel machine, non-common due date and varying processing time scheduling problem for jobs with varying quantities. This type of problem is common in manufacturing industries where batch production is significant. For example, in the automobile industry, items of varying quantities are typically assigned to different machines. The due dates for the jobs are different. The earliness and the tardiness penalties of the jobs also vary. Hence the manufacturing facility prioritizes the jobs, which are to be delivered earlier. This process results in completion of the jobs much earlier to its due date, which results in holding cost. Also certain products are completed later than its due date and so this incurs the lateness penalty. The parallel machines employed in all these cases need not be identical. Due to the huge investments involved in employing new technology machines, older technology machines are also employed in parallel. These old technology machines operate slowly when compared to new technology machines and so the processing times of the former are higher than the latter for the same job leading to a non-identical parallel machine environment.

Gupta (2000) had discussed the application of neural networks to select the best heuristic algorithm to solve a given scheduling problem. A two-stage hybrid flow shop with multiple identical parallel machines at the second stage is used as an example to discuss the process of selecting a

Corresponding Author: K. Raja, Department of Mechanical Engineering, Kumaraguru College of Technology, Coimbatore-641006, India Tel: 091-0422-2669401 Fax: 091-0422-2669406

scheduling heuristic through a neural-network approach. This research uses the genetic algorithm based approach for training the neural network and shows that the suggested neural network approach is quite effective and efficient for selecting the best heuristic algorithm for solving the scheduling problem. Agarwal *et al.* (2003) have proposed a new technique called Augmented Neural Networks (Aug NN) for solving the task-scheduling problem. This approach is a hybrid of the heuristic and the neural network approaches. While retaining all the advantages of the heuristic approach, AugNN incorporates learning, to get improved solutions iteratively. This new framework maps the problem structure to a neural network and utilizes domain specific knowledge for obtaining optimum solutions. The problem addressed is that of minimizing the makespan in scheduling n tasks on m machines where the tasks follow a precedence relation and task pre-emption is not allowed. Solutions obtained from AugNN using various learning rules are compared with six different commouly used heuristics. AugNN approach provides significant improvements over heuristic results. In just a few iterations, the gap between the lower bound and the obtained solution is reduced by as much as 58% for some heuristics, without any increase in computational complexity.

Park *et al.* (2000) assumed jobs to have sequence dependent set up times independent of the machine. The objective of scheduling was to find a sequence of the jobs, which minimizes the sum of weighted tardiness. The author has proposed an extension of the Apparent Tardiness Cost with Setups (ATCS) rule developed by Lee *et al.* (1997) which utilizes some look-ahead parameters for calculating the priority index of each job. Wbster and Azizoglu (2001) have addressed the problem of scheduling jobs with family setup times on identical parallel machines to minimize the total weighted flowtime. Two dynamic algorithms, a backward algorithm and a forward algorithm have been presented. Priore *et al.* (2003) have presented an approach to schedule jobs that employs machine learning. This is followed by the experimental study, which describes a new approach to determine new control attributes from the original ones. Akyol (2004) have considered the use of artificial neural network to model six different heuristic algorithms applied to the n job, m machine real flowshop scheduling problem with the objective of minimizing the makespan. The objective is to obtain six neural network models to be used for the prediction of the completion times for each job processed on each machine. This research aims at finding a multi-objective solution for a non-identical parallel machine envirounent. The problem under study is to schedule n jobs with non-common due dates and varying processing times on m non-identical parallel machines. The assumptions considered in this research are as follows.

- The processing time, set-up time and material handling time are known in advance.
- Job pre-emptions are not allowed.
- All machines and jobs are available at time zero.
- The jobs are independent of their sequence.

Three machines are available of which two machines are CNC machines and one machine is a conventional machine. All machines perform the same operations and the processing times of the conventional machines are higher than the CNC machines for the same job. The jobs numbering from 5 to 25 have to be processed on any one of the available parallel machines. The problem is to sequence the jobs such that the resulting sequence optimizes the multiple objectives. The objectives considered in this research are:

- Minimization of the earliness and the tardiness penalties
- Minimization of the completion time

Solution Methodology

Neural Network is an advanced heuristic technique that has been applied to difficult combinatorial problems in recent years to obtain good solutions. Moreover non-traditional algorithms when applied in a pure way generate satisfactory solutions in considering the multiple objectives. So Neural Network methodology has been adopted to generate optimal schedules for different set of jobs.

Neural Network

A neural network is a computational structure inspired by the study of biological neural processing. A layered feed forward neural network has layers, or subgroups of processing elements. A layer of processing element makes independent computations on data that it receives and passes the results to another layer. The next layer may in turn make its independent computations and pass on the results to yet another layer. Finally, a subgroup of one or more processing elements determines the output from the network. Each processing element makes its computation based upon a weighted sum of its inputs.

Output of a Neuron

Basically, the internal activation or raw output of a neuron in a neural network is a weighted sum of its inputs, but a threshold function is also used to determine the final value, or the output. When the input is 1, the neuron is said to fire and when it is 0, the neuron is considered not to have fired.

Training

Since the output may not be what is expected, the weights may need to be altered. Some rule then needs to be used to determine how to alter the weights. There should also be a criterion to specify when the process of successive modification of weights ceases. This process of updating the weights is called training.

Feedback

If a network is to be trained so it can recognize or identify some predetermined patterns, or evaluate some function values for given arguments, it would be important to have information feed back from the output neurons to neurons in some layer before that, to enable further processing and adjustment of weights on the connections.

Supervised Learning

The learning would be supervised if external criteria are used and matched by the network output and if not, the learning is unsupervised. This is one broad way to divide different neural network approaches.

Feed Forward Propagation Network

The feed forward back propagation network is a very popular model in neural networks. It does not have feedback connections, but errors are back propagated during training. Least mean squared error is used. Many applications can be formulated for using a feed forward back propagation network and the methodology has been a model for most multiplayer neural networks.

Training Algorithm

Step 0 : Initialize weight, (set of small random values).
Step 1 : While stopping condition are false, do steps 2-9.
Step 2 : For each training pair, do steps 3-8.

Feed Forward

Step 3 : Each input unit (Xi, I=1, 2.....p) receives input signal Xi and broadcast this signal to all units in the layer above (the hidden units)

Step 4 : Each hidden unit (Zj, j = 1, 2...p) sums its weighted input signal

$Z_{inj} = VO_j + \Sigma X_i V_{ij}$,

Applies its activation function to compute its output signal

$Zj = f(Z_{inj})$

And sends this signal to all units in the layers above (output units)

Step 5 : Each output unit (Y_k, k = 1, 2 and 3.... m) sums its weighted input signals.

$Y_{ink} = Wo_k + \Sigma Z_j W_{ij}$

And applies its activation function to compute its output signal

$Y_k = f(Y_{ink})$

Back Propagation of Error

Step 6 : Each output unit (Y_k, k = 1,2,3,....m) receives a target pattern corresponding to the input training

Pattern, computes its error information term.

$\ddot{A}_k = (t_k - Y_k) f(Y_{ink})$

Calculates its weights correction sum (used to update work later)

$W_{jk} = \alpha S_k Z_j$

Calculates its bias correction sum (used to update work later)

$Wo_k = \alpha S_k$

And sends SK units in the layer below.

Step 7 : Each hidden unit (Zj = j = 1, 2...p) Sums its delta inputs (from units in the layer above)

$S_{inj} = \Sigma S_k W_{jk}$

Update Weights and Biases

Step 8 : Each output unit (Y_k, k = 1, 2...m) Update its bias and weights (j = 0, 1, 2...p)

W_{jk} (new) = W_{jk} (old) + W_{jk}

Each hidden unit (Zj, j = 1, 2....p) updates its bias and weights (I = 0, 1, 2....n)

V_{ij} (new) = V_{ij}(old) + V_{ij}

Step 9 : Test for stopping condition.

C Implementation

In our C implementation of this network, we have the following arrays. We have separate arrays for input neurons and output neurons. The array A is for the input neurons. This array has weight and activation as data members. The B array is similar and is for the output neuron. The output neuron array has also a data member called output. There is another array for network included in B neuron. An instance of the network array is created with four input neurons. These four neurons are all connected with one output neuron.

COMPUTATIONAL RESULTS

The four input neurons represent the processing time of CNC machine, conventional machine, number of jobs, due date and quantities. The input neurons are multiplied with weights and these four

Table 1: Processing details for 25 jobs and 3 machines

Job	Processing time in M/C 1	Processing time in M/C 2	Due date	Quantity
1	3	6	3	682
2	7	9	7	1805
3	3	6	3	592
4	7	9	7	1747
5	2	4	2	291
6	4	7	4	1012
7	6	8	6	1722
8	5	8	5	1218
9	6	8	6	1519
10	6	9	6	1618
11	6	8	6	1567
12	6	9	6	1446
13	2	4	2	302
14	5	8	5	1307
15	5	7	5	1315
16	2	5	2	349
17	5	7	5	1224
18	6	9	6	1533
19	2	4	2	263
20	5	8	5	1124
21	6	8	4	1656
22	5	8	5	1401
23	2	4	6	480
24	2	5	6	357
25	3	5	2	675

Table 2: Weighted input for 25 jobs

Processing time in M/C 1	Processing time in M/C 2	Due date
0.6	0.6	1.5
1.4	0.9	3.5
0.6	0.6	1.5
1.4	0.9	3.5
0.4	0.4	1.0
0.8	0.7	2.0
1.2	0.8	3.0
1.0	0.8	2.5
1.2	0.8	3.0
1.2	0.9	3.0
1.2	0.8	3.0
1.2	0.9	3.0
0.4	0.4	1.0
1.0	0.8	2.5
1.0	0.7	2.5
0.4	0.5	1.0
1.0	0.7	2.5
1.2	0.9	3.0
0.4	0.4	1.0
1.0	0.8	2.5
1.2	0.7	2.0
1.0	0.8	2.5
0.4	0.4	3.0
0.4	0.5	3.0
0.6	0.5	1.0

neurons are sent to the hidden layers. The product of the processing time of neuron in input layer with the weightage factor of hidden layer gives weighted training data along with due date. Hence, many sequences are generated and the best sequence is obtained through the combined objective function value.

The details of the job processing times for 25 jobs and the weighted input have been shown in Table 1 and 2, respectively. The details of lateness of the jobs have been shown in Table 3. The

Table 3: Earliness (E) and Tardiness (T) for 25 jobs

Job	Processing time	Completion time	Due date	Lateness
5	0.4	0.4	2	E
13	0.4	0.8	2	E
16	0.5	1.3	2	E
19	0.4	1.7	2	E
23	0.4	2.1	6	E
24	0.5	2.6	6	E
1	0.6	3.2	3	T
3	0.6	3.8	3	T
25	0.5	4.3	2	T
6	0.6	4.9	4	T
8	1.0	5.9	5	T
14	0.8	6.7	5	T
15	1.0	7.7	5	T
17	0.7	8.4	5	T
20	1.0	9.4	5	T
22	0.8	10.2	5	T
7	1.2	11.4	6	T
9	1.2	12.6	6	T
10	0.9	13.5	6	T
11	1.2	14.7	6	T
12	1.2	15.9	6	T
18	0.9	16.8	6	T
21	1.2	18.0	4	T
2	0.9	18.9	7	T
4	1.4	21.3	7	T

optimum sequence obtained is: 5-13-16-19-23-24-1-3-25-6-8-14-15-17-20-22-7-9-10-11-12-18-21-2-4. The makespan is 7.2 min, number of early jobs is 6 and number of tardy jobs is 19.

Comparative Analysis

The results obtained through Neural Network technique have been compared with the existing H_1 algorithm used for parallel machine scheduling. In H_1algorithm, processing time and weighted processing time are considered as input. By determining the completion time of jobs, the lateness of the job can be found. Hence the performance measures such as makespan, earliness and tardiness can be optimized efficiently through Neural Network technique. It has been found that Neural Network results perform better compared to H_1 algorithm. For a 5 job problem, the sequence obtained through H_1 algorithm is 4-3-1-2-5 and the makespan obtained is 7 min and the number of tardy jobs is 5.

The same problem solved using H_1 algorithm has been solved using the proposed Neural Network technique. The optimum sequence obtained is 1-3-5-2-4. Optimum makespan is 1.26 min, the number of early jobs is 3 and number of tardy jobs is 4. The optimum makespan resulted from Neural Network technique is 5.74 min earlier than the heuristic method. Also the number of tardy jobs have been decreased to 3 from 5. Thus it is clear that the proposed method outperforms the heuristic technique.

CONCLUSIONS

In this research a neural network technique has been proposed for solving the parallel machine-scheduling problem. Quality Function Deployment have been used and the problem considered is that of optimizing the makespan, earliness and tardiness measures for scheduling n tasks on m identical machines. By varying the weights of make span, earliness and tardiness, the optimal performance measures are found and the best-weighted combination is identified. These optimal solutions are used for training the neural network. Through weights assigned to the links between tasks and machines and by adjusting the weights using an appropriate learning strategy, a significantly improved schedule is found using Neural Network. It has been observed that trained Neural Network models perform better when compared to existing heuristic methods.

REFERENCES

Agarwal, A., H. Pirkul and V.S. Jacob, 2003. Augmented neural networks for task scheduling. Eur. J. Operat. Res., 10: 112-123.

Akyol, D.E., 2004. Application of neural networks to heuristic scheduling algorithms. Comput. Ind. Eng., 46: 679-696.

Gupta, J.N.D., 2000. Selecting scheduling heuristics using neural network. INFORMS J. Comput., 12: 150-162.

Lee, Y.H., K. Bhaskaran and M. Pinedo, 1997. A heuristic to minimize the total weighted tardiness with sequence dependent set ups. IIE Trans., 29: 45-52.

Park, Y., S. Kim and Y. Hooul, 2000. Scheduling jobs on parallel machines applying neural network and heuristic rules. Comput. Ind. Eng., 38: 189-202.

Priore, P., D. Funte, R. Pino and J. Puento, 2003. Dynamic scheduling of flexible manufacturing systems using neural networks and inductive larning. Integrated Manufacturing Syst., 14: 160-168.

Wbster, S. and M. Azizoglu, 2001. Dynamic programming algorithms for scheduling parallel machines with family setup times. Comput. Operat. Res., 28: 127-137.