Matematika, 2001, Jilid 17, bil. 2, hlm. 63–70

A Neural Network for Common Due Date Job Scheduling Problem on Parallel Unrelated Machines

Abdelaziz Hamad, Bahrom Sanugi & Shaharuddin Salleh
Department of Mathematics
Universiti Teknologi Malaysia,
81310 UTM Skudai, Johor, Malaysia

Abstract This paper presents an approach for scheduling under a common due date on parallel unrelated machine problems based on artificial neural network. The objective is to allocate and sequence the jobs on the machines so that the total cost be minimized. This cost is composed of the total earliness and the total tardiness cost. Neural network is a suitable model in our study due to the fact that the problem is NP-hard. In our study, neural network has been proven to be effective and robust in generating near optimal solutions to the problem.

Keywords Neural networks, unrelated machines, scheduling.

Abstrak *Abstrak Kertas ini membincangkan pendekatan rangkaian neural bagi masalah penjadualan dengan tarikh akhir sepunya pada satu kumpulan mesin selari tak berkait. Objektif kajian ialah untuk mengagih dan menyusun kerja-kerja pada mesin supaya jumlah kos diminimumkan. Kos ini menpakan gubahan daripada jumlah kos awalan dan jumlah kos akhiran. Rangkaian neural adalah model yang sesuai dalam masalah ini kerana sifat masalah yang boleh dikategorikan sebagai NP-Sukar. Kajian ini membuktikan rangkaian neural adalah efektif dan kukuh dalam menghasilkan penyelesaian hampir optimum kepada masalah ini.*

Katakunci *Rangkaian neural, mesin tak terkait, penjadualan*

1 Introduction

Due to the industrial significance of the just-in-time philosophy, due date problems, i.e., scheduling problems where the due dates are given, have gained increasing attention in recent years. In such problems the jobs' due dates are fixed by the customer.

In this paper, we discuss the case of scheduling n independent jobs on m parallel-unrelated machines under a given common due date. We present a neural network procedure to solve this NP-hard problem. In our model, earliness is costed at the same rate for all jobs and so is tardiness, but the rates for earliness and tardiness differ. Each job requires a processing time, which differs from one machine to another. Schedules are assigned penalties, which are the sum of the costs related to earliness and tardiness of all jobs. By penalizing both the early and tardy completion of the jobs, costs related to inventory and customer satisfaction are recognized and taken into account. These costs are of a different nature. This is taken into consideration and incorporated into the model by allowing different weights for early and tardy completion. The objective is to schedule the jobs on the machines, so that the objective function be minimized. As the problem is NP-hard, we developed a neural network solution, which gives satisfactory results.

The total penalty function, which includes not only the total weighted earliness and tardiness but also the common due date, has been introduced by Panwalkar et al. [?] for the single machine case. After this, a lot of researchers have dealt with this kind of penalty function (Cheng & Gupta [?]). Alidaee and Panwalkar [?] provided an optimal solution for the problem of minimizing absolute lateness without a due date related penalty. Cheng & Gupta [?] examined the parallel identical machines problem when the due date is a decision variable. Since this problem is NP-hard, they presented a heuristic procedure providing efficient solutions. De et al. [?] have made some interesting comments on this problem and have introduced the use of inserted idle time in job schedules. Cheng and Chen [4] have extended the study of the parallel machine case by showing that the problem is NP-hard with both total and maximum penalty functions and by presenting a polynomial time algorithm for the special case of total penalty function where all jobs have equal processing times. In this paper we consider the parallel-unrelated machines problem when the common due date is given.

2 Problem Statement

Let N be the set of n independent jobs $J_1, J_2, J_3, , J_n$ to be processed on m unrelated parallel machines. The following notation shall be used,

S	schedule for the n jobs;
P_{ij}	processing time required by job J_i on the machine j;
d	common due date;
c_{ij}	the completion time of J_i on machine j;
E_{ij}	the earliness of J_i on machine j, which is equal to max $(0, d - c_{ij})$;
T_{ij}	the tardiness of J_i on machine j, which is equal to $\max(0, c_{ij} - d)$;
P_2, P_3,	the weights associated with earliness and tardiness respectively.

The problem considered is to schedule the n jobs on the m machines so that the objective function:

$$f(S) = \sum_{j=1}^{m} \sum_{i=1}^{n} (P_2 E_{ij} + P_3 T_{ij}) \tag{1}$$

be minimized with the following assumptions:

1. All jobs become available for machine processing simultaneously at time zero.
2. All processing times and common due date are deterministic and known before processing starts.

3. Job and machine pre-emption is not permitted.
4. The machines cannot process two or more jobs simultaneously.

3 A neural Network for Parallel Identical Machines

An artificial neural network is a collection of highly interconnected processing units that has the ability to learn and store patterns as well as to generalize when presented with new patterns. The 'learnt' information is stored in the form of numerical values, called weights that are assigned to the connections between the processing units of the network. A neural network usually consists of an input layer, one or more hidden layers and an output layer. Before the network is trained, the weights are assigned small, randomly determined values. Through a training procedure, such as backpropagation the network's weights are modified incrementally until the network is deemed to have learnt the relationship. This type of learning is a supervised type of learning. When a pattern is applied at the input layer, the stimulus is fed forward until final outputs are calculated at the output layer. The network's outputs are compared with the desired result for the pattern considered and the errors are computed. These errors are then propagated backwards through the network as feedback to the preceding layers to determine the changes in the connection weights to minimize the errors. A series of such input-output training examples is presented repeatedly until the total sum of the squares of these errors is reduced to an acceptable minimum. At this point the network is considered 'trained'. Data presented at the input layer of a trained network will result in values from the output layer consistent with the relationship learnt by the network from the training examples. The neural network that is proposed for the parallel unrelated machine common due date schedule problem is organized into three layers of processing units. There is an input layer of 16 units, a hidden layer, and an output layer that has 2 units. The number of units in the input and output layers is dictated by the specific representation adopted for the schedule problem. In the proposed representation, the input layer contains the information describing the problem in the form of a vector of continuous values.

The inputs is defined as follows:

$$\text{unit } 1 = \frac{p_{i1}}{M_{p_{i1}}} \tag{2}$$

$$\text{unit } 2 = \frac{p_{i2}}{M_{p_{i2}}} \tag{3}$$

$$\text{unit } 3 = \frac{d}{100} \tag{4}$$

$$\text{unit } 4 = \frac{SL_{i1}}{M_{i1}} \tag{5}$$

$$\text{unit } 5 = \frac{sl_{i2}}{M_{sl_{i2}}} \tag{6}$$

$$\text{unit } 6 = \frac{P_2}{10} \tag{7}$$

$$\text{unit } 7 = \frac{P_3}{10} \tag{8}$$

$$\text{unit } 8 = \frac{\overline{p}_{i1}}{M_{p_{i1}}} \tag{9}$$

$$\text{unit } 9 = \frac{\overline{p}_{i2}}{M_{p_{i2}}} \tag{10}$$

$$\text{unit } 10 = \frac{S\bar{l}_{i1}}{M_{Sl_{i1}}} \tag{11}$$

$$\text{unit } 11 = \frac{S\overline{L}_{i2}}{M_{sl_{i2}}} \tag{12}$$

$$\text{unit } 12 = 1 \tag{13}$$

$$\text{unit } 13 = \sqrt{\frac{\sum(p_{i1} - \overline{p}_{i1})^2}{n \times \overline{p}_{i1}^2}} \tag{14}$$

$$\text{unit } 14 = \sqrt{\frac{\sum(Sl_{i1} - S\overline{L}_{i1})^2}{n \times S\overline{L}_{i1}^2}} \tag{15}$$

$$\text{unit } 15 = \sqrt{\frac{\sum(p_{i2} - \overline{p}_{i2})^2}{n \times \overline{p}_{i2}^2}} \tag{16}$$

$$\text{unit } 16 = \sqrt{\frac{\sum(Sl_{i2} - S\bar{l}_{i2})^2}{n \times S\bar{l}_{i2}^2}} \tag{17}$$

where

Sl_{i1} slack for job J_i on the machine 1= $(d - p_{i1})$

Sl_{i2} slack for job J_i on the machine 2= $(d - p_{i2})$

Thus, each job is represented by a 16-input vector, which holds information particular to that job and in relation to the other jobs in the problem. The output unit assumes values that are in the range of 0.2-0.9, the magnitude being an indication of where the job represented at the input layer should desirably lie in the schedule. Low values suggest lead positions in the schedule; higher values indicate less priority and hence being positioned towards the end of the schedule. The number of units in the hidden layer is selected by trial and error during the training phase. The final network for two unrelated parallel machines has 14 units in its hidden layer and two on output units, and then it is described as a 16-14-2 network.

Table 1: (10 job, 2 unrelated machine)

Job_i	P_{i1}	P_{i2}	d	P_2	P_3	Sl_{i1}	Sl_{i2}
1	103	44	96	2	3	-7	52
2	29	71	96	2	3	67	25
3	67	106	96	2	3	29	-10
4	50	41	96	2	3	46	55
5	77	36	96	2	3	19	60
6	62	98	96	2	3	34	-2
7	41	40	96	2	3	55	56
8	58	16	96	2	3	38	80
9	95	99	96	2	3	1	-3
10	103	72	96	2	3	-7	24

Table 2: Problem representation for the example described in Table 1

Job_i	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	U_{11}	U_{12}
V1	1.00	0.42	0.96	-0.10	0.65	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V2	0.28	0.67	0.96	1.00	0.31	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V3	0.65	1.00	0.96	0.43	-0.13	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V4	0.49	0.39	0.96	0.69	0.69	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V5	0.75	0.34	0.96	0.28	0.75	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V6	0.60	0.92	0.96	0.51	-0.03	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V7	0.40	0.38	0.96	0.82	0.70	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V8	0.56	0.15	0.96	0.57	1.00	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V9	0.92	0.93	0.96	0.01	-0.04	0.20	0.30	0.67	0.59	0.41	0.42	1.00
V10	1.00	0.7	0.96	-0.10	0.30	0.20	0.30	0.67	0.59	0.41	0.42	1.00

Job	U_{13}	U_{14}	U_{15}	U_{16}	$\mathbf{Out}_1$	$\mathbf{Out}_2$
V1	0.36	0.89	0.50	0.93	0.00	**0.76**
V2	0.36	0.89	0.50	0.93	**0.43**	0.00
V3	0.36	0.89	0.50	0.93	**0.20**	0.00
V4	0.36	0.89	0.50	0.93	0.00	**0.34**
V5	0.36	0.89	0.50	0.93	0.00	**0.62**
V6	0.36	0.89	0.50	0.93	**0.67**	0.00
V7	0.36	0.89	0.50	0.93	0.00	**0.20**
V8	0.36	0.89	0.50	0.93	0.00	**0.48**
V9	0.36	0.89	0.50	0.93	**0.90**	0.00
V10	0.36	0.89	0.50	0.93	0.00	**0.90**

4 Example

To illustrate how the neural network is trained. Table **??** shows a 10-job problem and two unrelated parallel machines that serves as training example for neural network. The 10-jobs are converted first into their vector representations by using the set of equations (**??**-**??**). The result of this pre-processing stage is presented in Table **??** where the vectors $V_1 - V_{10}$ represent job numbers 1-10, respectively and the output for each of the input vectors is given in the two rightmost columns of Table **??**. To train the neural network, each vector with their output is presented individually at the input layer and output layer of the neural network. Training is considered complete after an average of 20000 cycles using a 16-14-2 configuration. A cycle is concluded after the network has been exposed once, in the course of the back propagation algorithm, to each one of the available training patterns. The trained neural network is used to find job schedule for any similar problem.

Table 3: (7 jobs, 2 unrelated machine)

Job_i	P_{i1}	P_{i2}	d	P_2	P_3	Sl_{i1}	Sl_{i2}
1	29	71	96	2	3	67	25
2	67	106	96	2	3	29	-10
3	50	41	96	2	3	46	55
4	77	36	96	2	3	19	60
5	62	98	96	2	3	34	-2
6	41	40	96	2	3	55	56
7	58	16	96	2	3	38	80

5 Numerical Example

Now the trained neural network is used to find the schedule for minimizing the cost function of equation 1. Table **??** shows a 7-job and two unrelated parallel machines and common due date. The 7- jobs are converted first into their vector representation by using the set of equations (**??**-**??**). The result of this pre-processing stage is presented in Table 4 where the vectors $V_1 - V_7$ represented job numbers 1-7, respectively. To solve the schedule problem, each vector is presented individually at the input layer of the neural network. A feed forward procedure of calculations generates a value that appears at the two-output unit for each of the 16 input vectors. The output computed by the neural network for each of the input vectors is given in the two end column of Table **??**.

Select the output1 and output 2 from the two rightmost columns of Table 4 and scheduling the jobs in the order of the increasing output values results in the job schedule for machine 1 and machine 2 respectively. $J_2 - J_1 - J_5$ for machine one with cost $= f(s_1) = 2(29) + 3(62) = 244$ and $J_6 - J_3 - J_7 - J_4$ for machine number two with cost $= f(s_2) = 2(56) + 2(15) + 3(1) + 3(37) = 256$. The total cost function equal $f(s_1) + f(s_2) = 244 + 256 = 500$.

6 Conclusion

The problem of job scheduling under a common due date on parallel unrelated machines has been examined in this paper. Given that all jobs share a common due date. Both total earliness and tardiness are penalized. The aim was to allocate and sequence the jobs on the machines such that the value of the objective function be as close to the optimal one as possible. Like in the case of identical machines, the problem of unrelated machines is NP-hard. A neural network as a heuristic solution has been proposed and a small system was designed in order to train neural network. On the basis of this small system, we observe that the neural network is very effective in obtaining near-optimal solutions. This represents a first attempt to use a neural network procedure to solve the common due date job scheduling problem on unrelated parallel machines.

Table 4: Problem representation for the example described in Table 3

Job	U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	U_{11}	U_{12}
V1	0.38	0.67	0.96	1.00	0.31	0.20	0.30	0.71	0.55	0.61	0.47	1.00
V2	0.87	1.00	0.96	0.43	-0.13	0.20	0.30	0.71	0.55	0.61	0.47	1.00
V3	0.65	0.39	0.96	0.69	0.69	0.20	0.30	0.71	0.59	0.61	0.47	1.00
V4	1.00	0.34	0.96	0.28	0.75	0.20	0.30	0.71	0.55	0.61	0.47	1.00
V5	0.81	0.92	0.96	0.51	-0.03	0.20	0.30	0.71	0.55	0.61	0.47	1.00
V6	0.53	0.38	0.96	0.82	0.70	0.20	0.30	0.71	0.55	0.61	0.47	1.00
V7	0.75	0.15	0.96	0.57	1.00	0.20	0.30	0.71	0.55	0.61	0.47	1.00

Job	U13	U14	U15	U16	Out1	Out2
V1	0.27	0.37	0.54	0.83	**0.40**	0.00
V2	0.27	0.37	0.54	0.83	**0.32**	0.00
V3	0.27	0.37	0.54	0.83	0.00	**0.42**
V4	0.27	0.37	0.54	0.83	0.00	**0.60**
V5	0.27	0.37	0.54	0.83	**0.60**	0.00
V6	0.27	0.37	0.54	0.83	0.00	**0.27**
V7	0.27	0.37	0.54	0.83	0.00	**0.48**

Acknowledgement The authors would like to thank Universiti Teknologi Malaysia (UTM) for providing their facilities for this research.

References

[1] Alidaee, B. Panwalkar, S.S *Single stage minimum absolute lateness problem with a common due date on non-identical machines.* Journal of Operational Research Society 44 (1993), 29-36.

[2] Baker, K. S., and. Scudder, G. D., *Sequencing with earliness and tardiness penalties*: A review, Oper. Res. 38 (1990), 22-36.

[3] Cheng, T.C.E, and M.C. Gupta, *Survey of scheduling research Involving due date Determination Decisions*, Euro. Journal Oper. Res., 38 (1989) 156-166.

[4] Cheng, T.C.E., Chen, Z.L.,*Parallel machine scheduling problems with earliness and tardiness penalties*, Journal of the Operational Research Society, 45 (1994), 685-695.

[5] De, P., Ghosh, J.B., Wells, C.E.,*On the multiple machine extension to a common due date assignment and scheduling problem*, Journal of the Operational Research Society 42 (1991),419-422.

[6] El-Bouri, A, Balakrishnan, S. and Popplewell, N., *Sequencing jobs on a Single Machine*: A neural Network Approach, Euro. Journal of Oper. Res., 126 (2000),474-490.

[7] Panwalkar, S.S., Smith, M.L., Seidmann, A.,*Common due date assignment to minimize total penalty for the one machine-scheduling problem*. Oper. Re. 30 (1982), 391-399.