International Journal of Management Science
Vol 3, No 2, November 1997

DYNAMIC SELECTION OF DISPATCHING RULES BY ARTIFICIAL NEURAL NETWORKS

JAE SIK LEE
School of Business Administration,
Ajou University, Suwon 442-749, Korea
Tel : +82-331-219-2719, E-mail:leejsk@www.ajou.ac.kr

ABSTRACT

Many heuristics have been developed in order to overcome the computational complexity of job shop problems. In this research, we develop a new heuristic by selecting four simple dispatching rules, i.e., SPT, LPT, SR and LR, dynamically as scheduling proceeds. The selection is accomplished by using artificial neural networks. As a result of testing on 50 problems, the makespan obtained by our heuristic is, on the average, 13.0% shorter than the longest makespan, and 0.4% shorter than the shortest makespan obtained by existing dispatching rules.

1. INTRODUCTION

For three decades, the job shop problem has held the attention of many researchers. The job shop problem is a production scheduling problem in which each one of n jobs must be processed in the time sequence on m machines, where a job consists of at most m operations. The assumptions of a job shop problem are summarized as follows [10]:

- A machine can process only one job at a time.
- An operation cannot be interrupted.
- The setup time for an operation is included in the processing time.
- The processing order of a job is given according to this job.
- The operation sequence on the machines are known.

The objective of this problem is to minimize some function of the completion times of the jobs subject to the constraints described above. In this research,

we choose the objective of minimizing the makespan, i.e., the time needed for processing all jobs. The job shop problem is known to be NP-hard, i.e., the time required to solve the problem to optimality increases exponentially as the problem size increases [4]. For very small problems, complete enumeration, branch and bound techniques or integer programming determine the optimal schedule [3], but for larger ones, efficient heuristics are necessary to solve or obtain near-optimal schedules [1].

Zanakis *et al.* [17] performed an extensive survey of journal publications on heuristic methods and applications in 442 articles published in 37 journals from 1968 to 1986. They classified them into twelve categories. About 25% of those articles is in the area of production and job shop scheduling. Therefore, it may fairly be stated that the scheduling problem has drawn much attention from many researchers, and it still is one of the most attractive problems.

In this research, we develop a new heuristic by selecting appropriate dispatching rules dynamically as a single job shop problem is being solved. The selection is accomplished by using artificial neural networks [8]. The structure of this article is as follows: In section 2, we first show the performance of four simple and popular dispatching rules, SPT, LPT, SR and LR on our randomly generated data. Then, we present a new heuristic that dynamically selects an appropriate dispatching rule for each operation as scheduling proceeds. In section 3, we employ artificial neural networks (ANN'S) as means of selecting appropriate dispatching rules. In this section, how we design, train and test the ANN is described. The computational experiments of the trained ANN are also presented in section 3. Finally, section 4 concludes this article.

2. DYNAMICALLY SELECTED DISPATCHING RULES

2.1 Performance of Existing Dispatching Rules

For many years, many researchers have been developing heuristic dispatching rules for job shop problems. A comprehensive survey of those rules is provided by Panwalkar and Iskander [11]. In our research, we choose four simple dispatching rules, SPT(shortest processing time), LPT(longest processing time), LR(longest remaining processing time) and SR(shortest remaining processing time).

The performance of a dispatching rule depends heavily on the structure of individual job shop problem. In other words, no single dispatching rule is dominant across all kinds of job shop problems. However, since no one can

judge which dispatching rule is suitable for a problem beforehand, many practicians tend to stick to one or two dispatching rules without knowing whether those dispatching rules are the best choices for their problems.

For our research, we randomly generated 50 problems with 3, 6, 9, 12 and 15 jobs, and 3, 6, 9, 12 and 15 machines, i.e., the pairs of jobs and machines are 3×3, 6×6, 9×9, 12×12 and 15×15; the integer processing times were randomly generated and uniformly distributed over the interval [1, 9]. We coded the scheduling program in Prolog, and the tests were performed on a 486 personal computer with 4MB RAM and a 80487 coprocessor. All test results of this research are presented in the Appendix. As shown in the dispatching rule columns in the Appendix, the variation in makespans yielded by different dispatching rules is apparent. For example, in one case of 9×9 problem Shop9-8, the worst schedule by SPT is even 38.2% longer than the shortest schedule by LR. Table 2.1 shows the number of the best schedules obtained by each dispatching rule on 50 test problems. The meaning of Vari% is described in the Appendix.

Table 2.1 Performance of Dispatching Rules by Problem Types

Job	Machine	SPT	LPT	SR	LR	Vari%
3	3	7	1	6	4	13.9%
6	6	3	2	4	4	13.1%
9	9	0	0	3	7	18.7%
12	12	0	0	2	8	14.7%
15	15	1	0	3	6	14.1%
Total		11	3	18	29	14.9%

As seen in Table 2.1, LR performs best, then SR, SPT and LPT in that order. On the average of all 50 problems, the worst schedule is 14.9% longer than the best, i.e., the shortest schedule. Therefore, it is not a good idea to stick to one dispatching rule regardless of the structure or intrinsic nature of the problem. In the next subsection, we describe how we can get a better schedule by dynamically selecting the best dispatching rules as jobs are scheduled.

2.2 Selection of Dispatching Rule for Each Operation

In this section, we describe the heuristic called the *Dynamically Selected Dispatching Rule* (DSDR) developed in our research. For each operation, our DSDR heuristic first checks to select one of the four dispatching rules, SPT, LPT, SR and LR, that yields the best schedule, i.e., the shortest makespan.

Then, the DSDR heuristic schedules the operations of all jobs according to the selected dispatching rule. Let us call the machine on which more than one job are to be scheduled in current operation the *B_machine* which stands for *Busy machine*. The DSDR heuristic selects the best dispatching rule efficiently by trying the four dispatching rules only on the *B_machine*. The DSDR heuristic yields a schedule in the following way:

WHILE all operations are not scheduled DO
WHILE all jobs are not scheduled DO

1. Identify *B_machines*.
2. Schedule the jobs queued on the *B_machines* using SPT, LPT, SR and LR.
3. Calculate the makespan of the resulting partial schedule.
4. Select the dispatching rule that gives the shortest makespan.
   If more than two rules tie,
   then select a rule in the order, LR, SR, SPT and LPT.
5. Schedule all jobs on all machines according to the selected dispatching rule.

END WHILE (jobs)
END WHILE (operations)

As described in the above procedure, if more than two rules yield the same values of makespan, the DSDR heuristic selects a dispatching rule in the order, LR $\rightarrow$ SR $\rightarrow$ SPT $\rightarrow$ LPT. For example, if SPT and LPT yield the same values of makespan, the DSDR heuristic selects SPT. The order of selection was determined according to the performance comparison presented in Table 2.1.

The DSDR heuristic was tested on our 50 test problems. The sequence of the selected dispatching rules, for example, for the problem Shop9_5 is (LR, SPT, LR, SR, LR, SPT, LR, LR, LR), and the resulting makespan is 86 which is better than that of any individual dispatching rule. In other words, the DSDR heuristic yields the best schedule for the problem Shop9_5. The performance of the DSDR heuristic is also shown in the Appendix. On the average of all 50 test problems, by the DSDR heuristic, the makespan is decreased 13.7% from the worst makespan obtained by individual dispatching rules, and 1.2% from the best makespan obtained by individual dispatching rules. Therefore, it may be stated that the DSDR heuristic performs as good as the best dispatching rule among SPT, LPT, SR and LR does.

The DSDR heuristic, however, does not always yield the best schedules. Out of 50 problems, the number of cases when the DSDR schedules are the shortest is 40; the number of cases when the DSDR schedules are at least the

Table 2.2 Performance of DSDR Heuristic by Problem Types

Job	Machine	SPT	LPT	SR	LR	DSDR	MinD%	MaxD%
3	3	6	1	5	4	7	-2.3%	9.3%
6	6	2	1	2	2	8	1.5%	12.7%
9	9	0	0	1	5	7	0.5%	15.9%
12	12	0	0	1	1	9	3.7%	15.9%
15	15	0	0	1	3	9	2.6%	14.6%
Total		8	2	10	15	40	1.2%	13.7%

second best is 50; there is no case when the DSDR schedules are the worst. Table 2.2 shows the number of the best schedules obtained by the four dispatching rules and the DSDR heuristic. The meanings of MinD% and MaxD% are described in the Appendix.

Even though we tried each dispatching rule only on the jobs queued on *B_machines*, the rule selection routine in the DSDR is certainly a brute force procedure. For each operation, the DSDR heuristic actually schedules the jobs according to each dispatching rule, and then selects the best after comparing the four values of makespan of resulting partial schedules. If the number of *B_machines* or the number of jobs queued on *B_machines* increases, the computation time of the DSDR heuristic will increase, and most of the computation time will be spent on the rule selection routine. To overcome this inefficiency, we will employ artificial neural networks (ANN'S) for selecting appropriate dispatching rules. This subject is discussed in the next section.

3. EMPLOYMENT OF ANN FOR DISPATCHING RULE SELECTION

3.1 Artificial Intelligence Application in Scheduling Problem

The artificial intelligence (AI) techniques used in scheduling problems are mainly in the category of rule-based inference, i.e., expert systems [16]. Expert systems for scheduling problems are found, e.g., in [6, 7, 9, 12, 14]. Since they are rule-based expert systems, the knowledge about scheduling must be represented in IF-THEN form. However, one of the main disadvantages of expert systems approach is the difficulty in knowledge acquisition let alone the fact that the knowledge represented in IF-THEN form is very inflexible. Representing the heuristic knowledge of complex scheduling problem in IF-THEN form is a very complicated and difficult task. Therefore, we chose the

ANN technique to extract the heuristic knowledge, even though the knowledge acquired in ANN is not explicit.

The research presented in this paper is in the category of Machine Learning. Machine Learning is the most recent application of AI techniques in scheduling problems. Arizono *et al.* [2] developed a heuristic for a single machine scheduling problem by using the Gaussian machine model which is one of the stochastic neural network models. They tested their heuristic on 50 problems, each consists of 10 jobs and a single machine. The result was that the average approximate rate of obtained schedules to optimal solutions was 0.9. Kadaba *et al.* [5] developed XROUTE, a software system that yields near-optimal results for vehicle routing and scheduling problems. They tested their system on 25 randomly generated problems and compared the solutions with those obtained by four existing heuristics. The computational results showed that, in all 25 problems, XROUTE yielded the best solutions. Shaw *et al.* [15] developed a scheduling approach called PDS (Pattern-Directed Scheduling) based on the inductive learning process called ID3 algorithm [13]. They tested the PDS approach on 69 different settings of manufacturing pattern. The number of cases when the PDS solutions were worse than the best solutions obtained by dispatching rules was 10.

3.2 Artificial Neural Network as Dispatching Rule Selector

As discussed in section 2.2, by dynamically selecting the best dispatching rules, i.e., by the DSDR heuristic, we can obtain better schedules. However, the DSDR heuristic certainly takes longer than individual dispatching rule. After having examined the rule selection patterns in the DSDR heuristic, we found that there must be some structure in a partial schedule that lends itself to fit for a certain dispatching rule. In other words, there must be some relationship between a partial schedule and the best dispatching rule for the jobs to be scheduled in the next operation. We employed an ANN approach to extract that relationship. We shall call this ANN the *ANN_DRS* (**A**rtificial **N**eural **N**etwork as **D**ispatching **R**ule **S**elector). We designed three ANN_DRS's, i.e., ANN_DRS3, ANN_DRS4 and ANN_DRS5. ANN_DRS3 is for the case when the number of jobs queued on *B_machine* is three. The other two ANN_DRS's are for the cases when the number is four and five, respectively.

In this section, we explain the structure of ANN_DRS3 and how we trained and tested it. The other two ANN_DRS's can be explained in the same manner. The input values for ANN_DRS3 are determined as follows. For operation K, a job J to be scheduled on a *B_machine* can be represented by three values as follows:

- J_d : The time when the $(K-1)^{th}$ operation of the job J is finished.
- J_p : The processing time of the K^{th} operation of the job J.
- J_r : The remaining processing time of the job J.

The status of the *B-machine* M is represented by a single value as follows:

- M_a: The available point of time of machine M, i.e., the latest finishing time of the $(K-1)^{th}$ operation of any job on machine M.

The data for input layer for training and testing the ANN_DRS3 are generated as follows:

- Let us denote J_{di} the value of J_d of the i^{th} job. J_{d1} is randomly generated from interval [1, 150]. J_{d2} is from interval [J_{d1} , $J_{d1}+20$], and J_{d3} is from interval [J_{d2} , $J_{d2}+20$]. All J_d's are integers.
- Let us denote $\overline{J_d}$ the average value of J_d's. M_a is integer and randomly generated from interval [$\overline{J_d}-20$, $\overline{J_d}+20$].
- J_p's are integer and randomly generated from interval [1, 9].
- J_r's are not generated. Only the order of jobs, i.e., J_{Oi}, with respect to the remaining processing time is randomly generated.

The input layer of ANN_DRS3 consists of ten Processing Elements (PE's), i.e., nine PE's for representing three jobs and one PE for *B-machine*. The output layer consists of four PE's, each represents the selection decision of single dispatching rule. If SPT is selected, the data for the output layer is a set of four numbers, (1,0,0,0); LPT, (0,1,0,0); SR, (0,0,1,0); LR, (0,0,0,1). The data for output layer is obtained by actually performing the DSDR heuristic with the data for input layer. The number of PE's in hidden layer is set to six. The transfer function of the input layer is linear. The transfer functions of the hidden and the output layer are Sigmoid functions. The learning process is performed using backpropagation algorithm. The structure of ANN_DRS3 is depicted in Figure 3.1.

We generated 100 pairs of data, i.e., input data and the matching output data, for each ANN_DRS. Among these data, 70 pairs were used for training, and 30 pairs were used for testing the ANN_DRS. Table 3.1 shows the

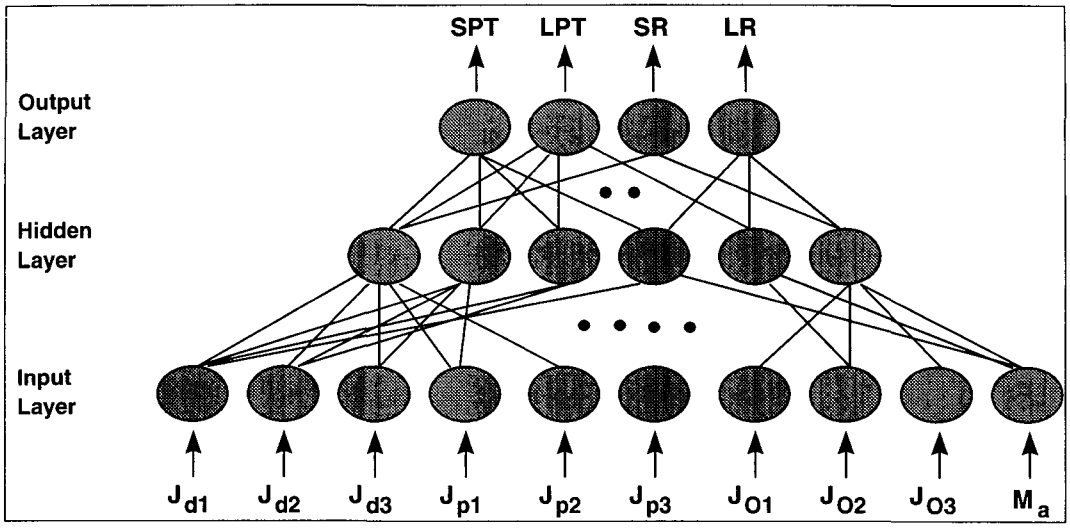

Figure 3.1 Structure of ANN_DRS3

accuracy rate of the trained ANN_DRS's for various settings of training times. For example, after we trained the ANN_DRS3 for 40000 times, the trained ANN_DRS3 selected the best dispatching rules with full accuracy for the 70 pairs of training data. However, for the 30 pairs of test, i.e., holdout data, it selected the best dispatching rules with the accuracy rate of 90%. The bold-faced figures represent the accuracy rates of the selected ANN_DRS's. For example, for ANN_DRS3, the network trained for 30000 times was selected.

Table 3.1 Accuracy Rate of ANN_DRS's under Various Training Times

Training Time	ANN_DRS3		ANN_DRS4		ANN_DRS5	
	Train(%)	Test(%)	Train(%)	Test(%)	Train(%)	Test(%)
20000	99	90	89	82	86	72
25000	100	89	**90**	**84**	**89**	**72**
30000	**100**	**90**	91	81	89	69
35000	100	90	86	85	90	71
40000	100	90	88	84	90	70

As shown in Table 3.1, the performances of the trained ANN_DRS's are acceptable. However, the performance of the trained ANN_DRS is getting worse as the number of jobs queued on *B_machine* increases. It implies that the lookahead capability of ANN_DRS is diminishing as the number of jobs to be scheduled increases.

4. DISPATCHING RULES SELECTED BY ARTIFICIAL NEURAL NETWORKS

By incorporating ANN_DRS3, ANN_DRS4, ANN_DRS5 into one procedure, we develop a new heuristic which we shall call *DRANN* (**D**ispatching **R**ules selected by **A**rtificial **N**eural **N**etworks). For each operation, the DRANN heuristic first checks each machine whether there are any jobs to be scheduled, i.e., *queued jobs*. The DRANN heuristic consists of several scheduling routines each of which is activated depending on the number of queued jobs.

If there are two queued jobs, then schedule them using a simple rule such as SPT. If there are three, four or five queued jobs, then select a dispatching rule using ANN_DRS's and schedule them according to the selected dispatching rule. If there are more than five queued jobs, first sort them in the descending order of remaining processing time. Then for the first three jobs, select a dispatching rule using ANN_DRS3 and schedule them according to the selected dispatching rule. With the remaining queued jobs, the DRANN heuristic performs an appropriate scheduling routine all over again. This procedure continues until all operations are scheduled.

The performances of the DRANN heuristic on our 50 test problems are also shown in the Appendix. The results shown in the Appendix indicate that the DRANN heuristic performs as good as the DSDR heuristic does. Table 4.1 shows the performance of the DRANN compared to the DSDR. Inc% and Dev% of each problem type are calculated as follows:

T : Set of problem types, i.e., $T = \{ 3\times3,\ 6\times6,\ 9\times9,\ 12\times12,\ 15\times15 \}$

I : Set of problem numbers, i.e., $I = \{ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10 \}$

$$\text{Inc\%}_t = \frac{1}{10}\sum_{i\in I}\frac{\text{DRANN}_i - \text{DSDR}_i}{\text{DSDR}_i}\times 100, \qquad t \in T$$

$$\text{Dev\%}_t = \frac{1}{10}\sum_{i\in I}\frac{|\ \text{DRANN}_i - \text{DSDR}_i\ |}{\text{DSDR}_i}\times 100, \qquad t \in T$$

Table 4.1 Performance of DRANN Compared to DSDR by Problem Types

Job	Machine	Inc%	Dev%
3	3	-0.2%	4.5%
6	6	1.0%	3.0%
9	9	1.0%	3.3%
12	12	1.0%	3.0%
15	15	1.9%	4.6%
Average		0.9%	3.7%

As shown in Table 4.1, the DRANN makespan is, on the average of each problem type, slightly longer than, but within the range of 5% of the DSDR makespan. However, since the DRANN heuristic employs ANN for selecting dispatching rules, it takes much shorter than the DSDR heuristic to finish scheduling.

The comparisons of computation times of DSDR and DRANN for each type of problems are presented in Table 4.2. In order to differentiate the computation times of the two procedures more vividly, they were measured using 33 MHz 386 PC with 80387 Coprocessor, i.e., a slower machine.

Table 4.2 Comparisons of Computation Times of DSDR and DRANN

unit : seconds

Type / Number	3×3		6×6		9×9		12×12		15×15	
	DSDR	DRANN	DSDR	DRANN	DSDR	DRANN	DSDR	DRANN	DSDR	DRANN
1	0.61	0.21	1.64	0.38	4.28	0.93	9.17	2.36	19.85	3.65
2	0.44	0.01	1.81	0.61	5.32	1.01	10.07	2.12	19.46	3.42
3	0.39	0.11	1.76	0.59	4.89	1.39	11.97	2.47	18.82	3.95
4	0.44	0.11	1.55	0.38	4.44	1.17	11.15	2.44	22.11	4.11
5	0.43	0.12	2.03	0.59	5.34	1.11	9.07	2.34	24.09	4.09
6	0.38	0.11	1.80	0.55	4.24	1.00	10.25	2.15	21.78	3.73
7	0.49	0.06	1.86	0.31	4.19	1.15	13.16	2.62	20.07	3.30
8	0.50	0.16	1.85	0.65	5.17	1.30	11.95	2.44	20.48	4.73
9	0.38	0.10	1.92	0.60	5.07	1.30	10.73	2.36	20.24	4.09
10	0.37	0.05	2.02	0.72	5.00	1.60	9.52	2.40	22.33	4.11
Average	0.44	0.10	1.82	0.54	4.79	1.20	10.70	2.37	20.92	3.92

As shown in Table 4.2, the computation time of DRANN is much shorter than that of DSDR. On the average, the computation time of DRANN is 24% of that of DSDR.

Both of the DSDR and the DRANN heuristics consist of two parts: One part is to select an appropriate dispatching rule, and the other part is to schedule the jobs according to the selected dispatching rule. Table 4.3 and Table 4.4 show the times spent on the selecting part and the scheduling part of the DSDR and the DRANN heuristics, respectively.

Table 4.3 DSDR Procedure : Time spent on Selection and Scheduling*

unit : seconds

Type / Number	3×3		6×6		9×9		12×12		15×15	
	Select	Schedule	Select	Schedule	Select	Schedule	Select	Schedule	Select	Schedule
1	0.49	0.12	1.30	0.34	3.01	1.27	6.37	2.80	12.30	7.55
2	0.38	0.06	1.42	0.39	3.84	1.48	7.04	3.03	11.70	7.76
3	0.28	0.11	1.43	0.33	3.50	1.39	8.02	3.95	11.24	7.58
4	0.32	0.12	1.01	0.54	3.08	1.36	7.81	3.34	14.46	7.65
5	0.37	0.06	1.63	0.40	4.12	1.22	6.26	2.81	16.17	7.92
6	0.33	0.05	1.36	0.44	2.97	1.27	6.64	3.61	14.72	7.06
7	0.44	0.05	1.49	0.37	3.03	1.16	8.72	4.44	12.29	7.78
8	0.44	0.06	1.43	0.42	3.79	1.38	8.06	3.89	13.90	6.58
9	0.32	0.06	1.43	0.49	3.75	1.32	7.42	3.31	12.82	7.42
10	0.22	0.15	1.52	0.50	3.73	1.27	6.72	2.80	13.49	8.84
Average	0.36	0.08	1.40	0.42	3.48	1.31	7.04	3.40	13.31	7.61

Table 4.4 DRANN Procedure : Time spent on Selection and Scheduling*

unit : seconds

Type / Number	3×3		6×6		9×9		12×12		15×15	
	Select	Schedule	Select	Schedule	Select	Schedule	Select	Schedule	Select	Schedule
1	0.11	0.10	0.05	0.33	0.17	0.76	0.99	1.37	1.22	2.43
2	0.00	0.01	0.11	0.50	0.38	0.63	1.04	1.08	1.12	2.30
3	0.00	0.11	0.22	0.37	0.60	0.79	0.99	1.48	1.31	2.64
4	0.00	0.11	0.11	0.27	0.39	0.78	1.05	1.39	1.38	2.73
5	0.00	0.12	0.22	0.37	0.38	0.73	0.84	1.50	1.71	2.38
6	0.00	0.11	0.33	0.22	0.44	0.56	0.93	1.22	1.23	2.50
7	0.00	0.06	0.11	0.20	0.50	0.65	0.93	1.69	1.04	2.26
8	0.11	0.05	0.22	0.43	0.56	0.74	1.03	1.41	1.77	2.96
9	0.00	0.10	0.22	0.38	0.44	0.86	0.77	1.59	1.25	2.84
10	0.00	0.05	0.22	0.50	0.87	0.73	0.92	1.48	1.42	2.69
Average	0.02	0.08	0.18	0.36	0.47	0.73	0.95	1.42	1.35	2.57

* Select : Time spent on selection of dispatching rule
Schedule : Time spent on scheduling
Select + Schedule gives the total time shown in Table 4.2.

As we have expected, most of the DSDR computation time was spent on the selecting part. On the average, 72.49% of the computation time was spent on the selecting part. For the DRANN heuristic, however, 33.40% of the computation time was spent on the selecting part.

5. CONCLUSION

In order to smooth out the variation among the makespans obtained by the four simple dispatching rules, i.e., SPT, LPT, SR and LR, we develop a new heuristic in which one of the four dispatching rules is selected and used, operation by operation, according to the structure of the individual job shop problem. The procedure of dispatching rule selection must be very time-consuming if we perform actual scheduling to select the best dispatching rule. In this research, however, we successfully incorporate the dispatching rule selection procedure in our heuristic minimally affecting the overall computation time by using artificial neural networks.

The performance of our heuristic is quite acceptable. As a result of testing on 50 randomly generated problems, the makespan obtained by our heuristic is, on the average, 0.4% shorter than the shortest makespan, and 13.0% shorter than the longest makespan obtained by the four dispatching rules. In this research, we showed that better makespan can be obtained by using several dispatching rules in solving a single job shop problem. There are so many dispatching rules, and the combination of dispatching rules must be limitless. Therefore, there may be another combination of dispatching rules that yields better makespan than our heuristic does. The structure of ANN may also be designed differently. In particular, we set the number of PE's in hidden layer to six. However, by varying the number of PE's in hidden layer, we could design an ANN that yields better accuracy rates. We leave these subjects for further research.

Acknowledgement
This research was supported by the 1996 Research Fund at Ajou University.

REFERENCES

[1] Adams, J., E. Balas and D. Zawack, "The Shifting Bottleneck Procedure for Job Shop Scheduling," *Management Sci.* 34, 3 (1988), 391-401.

[2] Arizono, I., A. Yamamoto and H. Ohta, "Scheduling for Minimizing Total Actual Flow Time by Neural Networks," *Int. J. of Prod. Res.* 30, 3 (1992), 503-511.

[3] Carlier, J. and E. Pinson, "An Algorithm for Solving the Job-Shop Problem," *Management Sci.* 35, 2 (1989), 164-176.

[4] Garey, M. R., D. S. Johnson and R. R. Sethi, "The Complexity of Flowshop and Jobshop Scheduling," *Mathematics of Oper. Res.* 1 (1976),

117-129.

[5] Kadaba, N., K. E. Nygard and P. L. Juell, "Integration of Adaptive Machine Learning and Knowledge-Based Systems for Routing and Scheduling Applications," *Expert Systems with Appl.* 2 (1991), 15-27.

[6] Kanet, J. J. and H. H. Adelsberger, "Expert Systems in Production Scheduling," *Euro. J. Oper. Res.* 29 (1987), 51-59.

[7] Kerr, R. M. and R. V. Ebsary, "Implementation of an Expert System for Production Scheduling," *Euro. J. Oper. Res.* 33 (1988), 17-29.

[8] Kosko, B., *Neural Networks and Fuzzy Systems*, Prentice-Hall Int'l Eds. (1992).

[9] Kusiak, A. and M. Chen, "Expert Systems for Planning and Scheduling Manufacturing Systems," *Euro. J. Oper. Res.* 34 (1988), 113-130.

[10] McCarthy, B. L. and J. Liu, "Addressing the Gap in Scheduling Research: A Review of Optimization and Heuristic Methods in Production Scheduling," *Int. J. Prod. Res.* 31, 1 (1993), 59-79.

[11] Panwalkar, S. S. and W. Iskander, "A Survey of Scheduling Rules," *Oper. Res.* 25, 1 (1977), 45-61.

[12] Pierreval, H., "Expert System for Selecting Priority Rules in Flexible Manufacturing Systems," *Expert Systems with Appl.* 5 (1992), 51-57.

[13] Quinlan, J., "Discovering Rules by Induction from Large Collection of Examples," in D. Michie (ed.), *Expert Systems in the Micro Electronic Age*, Edinburgh Univ. Press, Edinburgh, Scotland, 1979.

[14] Randhawa, S. U. and E. D. McDowell, "An Investigation of the Applicability of Expert Systems to Job Shop Scheduling," *Int. J. Man-Machine Studies* 32 (1990), 203-213.

[15] Shaw, M. J., S. Park and N. Raman, "Intelligent Scheduling with Machine Learning Capabilities: The Induction of Scheduling Knowledge," *IIE Transactions* 24, 2 (1992), 156-168.

[16] Waterman, D. A., *A Guide to Expert Systems*, Addison-Wesley Pub. Co. (1986).

[17] Zanakis, S. H., J. R. Evans and A. A. Vazacopoulos, "Heuristic Methods and Applications: A Categorized Survey," *Euro. J. Oper. Res.* 43 (1989), 88-110.

Appendix

	SPT	LPT	SR	LR		DSDR			DRANN		
					Vari%		MinD%	MaxD%		MinD%	MaxD%
Shop3_1	28	31	28	31	10.7%	31	-10.7%	0.0%	28	0.0%	9.7%
Shop3_2	27	30	34	30	25.9%	27	0.0%	20.6%	27	0.0%	20.6%
Shop3_3	25	24	25	24	4.2%	24	0.0%	4.0%	25	-4.2%	0.0%
Shop3_4	32	33	32	33	3.1%	31	3.1%	6.1%	32	0.0%	3.0%
Shop3_5	28	29	28	29	3.6%	29	-3.6%	0.0%	28	0.0%	3.4%
Shop3_6	22	23	22	23	4.5%	22	0.0%	4.3%	22	0.0%	4.3%
Shop3_7	29	30	27	22	36.4%	22	0.0%	26.7%	22	0.0%	26.7%
Shop3_8	46	46	37	35	31.4%	35	0.0%	23.9%	40	-14.3%	13.0%
Shop3_9	25	27	25	25	8.0%	25	0.0%	7.4%	25	0.0%	7.4%
Shop3_10	26	29	26	29	11.5%	29	-11.5%	0.0%	26	0.0%	10.3%
Average	28.8	30.2	28.4	28.1	13.9%	27.5	-2.3%	9.3%	27.5	-1.8%	9.9%
Shop6_1	72	85	70	85	21.4%	64	8.6%	24.7%	66	5.7%	22.4%
Shop6_2	59	63	59	57	10.5%	54	5.3%	14.3%	55	3.5%	12.7%
Shop6_3	65	70	65	58	20.7%	58	0.0%	17.1%	54	6.9%	22.9%
Shop6_4	50	52	50	57	14.0%	50	0.0%	12.3%	50	0.0%	12.3%
Shop6_5	64	58	64	59	10.3%	57	1.7%	10.9%	59	-1.7%	7.8%
Shop6_6	53	52	54	50	8.0%	50	0.0%	7.4%	54	-8.0%	0.0%
Shop6_7	47	47	52	48	10.6%	48	-2.1%	7.7%	48	-2.1%	7.7%
Shop6_8	76	83	76	83	9.2%	74	2.6%	10.8%	73	3.9%	12.0%
Shop6_9	65	65	60	58	12.1%	56	3.4%	13.8%	58	0.0%	10.8%
Shop6_10	72	71	63	66	14.3%	66	-4.8%	8.3%	65	-3.2%	9.7%
Average	62.3	64.6	61.3	62.1	13.1%	57.7	1.5%	12.7%	58.2	0.5%	11.8%
Shop9_1	92	98	88	96	11.4%	85	3.4%	13.3%	79	10.2%	19.4%
Shop9_2	101	112	105	94	19.1%	96	-2.1%	14.3%	101	-7.4%	9.8%
Shop9_3	110	118	115	100	18.0%	100	0.0%	15.3%	97	3.0%	17.8%
Shop9_4	77	77	85	74	14.9%	77	-4.1%	9.4%	78	-5.4%	8.2%
Shop9_5	99	107	98	91	17.6%	86	5.5%	19.6%	88	3.3%	17.8%
Shop9_6	95	82	94	80	18.8%	78	2.5%	17.9%	82	-2.5%	13.7%
Shop9_7	89	92	97	84	15.5%	84	0.0%	13.4%	85	-1.2%	12.4%
Shop9_8	105	96	100	76	38.2%	76	0.0%	27.6%	80	-5.3%	23.8%
Shop9_9	97	93	87	91	11.5%	82	5.7%	15.5%	83	4.6%	14.4%
Shop9_10	134	123	110	122	21.8%	117	-6.4%	12.7%	115	-4.5%	14.2%
Average	99.9	99.8	97.9	90.8	18.7%	88.1	0.5%	15.9%	88.8	-0.5%	15.1%

Vari% = (Longest DR Makespan - Shortest DR Makespan) / Shortest DR Makespan × 100%
MinD% = (Shortest DR Makespan - Our Heuristic Makespan) / Shortest DR Makespan × 100%
MaxD% = (Longest DR Makespan - Our Heuristic Makespan) / Longest DR Makespan × 100%
where DR is Dispatching Rule, Our Heuristic is DSDR or DRANN.

Appendix(continued)

	SPT	LPT	SR	LR		DSDR			DRANN		
					Vari%		MinD%	MaxD%		MinD%	MaxD%
Shop12_1	148	159	136	155	16.9%	116	14.7%	27.0%	121	11.0%	23.9%
Shop12_2	132	124	129	118	11.9%	118	0.0%	10.6%	112	5.1%	15.2%
Shop12_3	167	168	173	155	11.6%	151	2.6%	12.7%	152	1.9%	12.1%
Shop12_4	131	129	144	128	12.5%	116	9.4%	19.4%	123	3.9%	14.6%
Shop12_5	136	140	140	130	7.7%	128	1.5%	8.6%	127	2.3%	9.3%
Shop12_6	137	150	145	132	13.6%	131	0.8%	12.7%	133	-0.8%	11.3%
Shop12_7	160	154	147	132	21.2%	127	3.8%	20.6%	129	2.3%	19.4%
Shop12_8	125	142	121	133	17.4%	125	-3.3%	12.0%	120	0.8%	15.5%
Shop12_9	147	138	141	130	13.1%	123	5.4%	16.3%	125	3.8%	15.0%
Shop12_10	144	139	124	119	21.0%	116	2.5%	19.4%	121	-1.7%	16.0%
Average	142.7	144.3	140	133.2	14.7%	125.1	3.7%	15.9%	126.3	2.9%	15.2%
Shop15_1	161	165	151	145	13.8%	146	-0.7%	11.5%	137	5.5%	17.0%
Shop15_2	194	177	172	175	12.8%	162	5.8%	16.5%	159	7.6%	18.0%
Shop15_3	179	185	170	163	13.5%	162	0.6%	12.4%	154	5.5%	16.8%
Shop15_4	205	201	200	174	17.8%	174	0.0%	15.1%	182	-4.6%	11.2%
Shop15_5	167	198	172	161	23.0%	161	0.0%	18.7%	172	-6.8%	13.1%
Shop15_6	161	189	164	166	17.4%	158	1.9%	16.4%	163	-1.2%	13.8%
Shop15_7	170	170	169	156	9.0%	150	3.8%	11.8%	159	-1.9%	6.5%
Shop15_8	183	187	179	177	5.6%	159	10.2%	15.0%	167	5.6%	10.7%
Shop15_9	162	156	146	156	11.0%	146	0.0%	9.9%	156	-6.8%	3.7%
Shop15_10	187	174	159	163	17.6%	152	4.4%	18.7%	151	5.0%	19.3%
Average	176.9	180.2	168.2	163.6	14.1%	157	2.6%	14.6%	160	0.8%	13.0%

Vari% = (Longest DR Makespan - Shortest DR Makespan) / Shortest DR Makespan × 100%
MinD% = (Shortest DR Makespan - Our Heuristic Makespan) / Shortest DR Makespan × 100%
MaxD% = (Longest DR Makespan - Our Heuristic Makespan) / Longest DR Makespan × 100%
where DR is Dispatching Rule, Our Heuristic is DSDR or DRANN.