A Practical Method for Multi-Objective Scheduling through Soft Computing Approach*

Yoshiaki SHIMIZU** and Yasutsugu TANAKA**

Due to diversified customer demands and global competition, scheduling has been increasingly notified as an important problem-solving in manufacturing. Since the scheduling is considered at stage close to the practical operation in production planning, flexibility and agility in decision making should be most important in real world applications. In addition, since the final goal of such scheduling has many attributes, and their relative importance is likely changed depending on the decision environment, it is of great significance to derive a flexible scheduling through plain multi-objective optimization method. To derive such a rational scheduling, in this paper, we have applied a novel multi-objective optimization named $MOON^{2R}$ ($MOON^2$ of radial basis function) by incorporating with simulated annealing as a solution algorithm. Finally, illustrative examples are provided to outline and verify the effectiveness of the proposed method.

***Key Words*:** Multi-Objective Optimization, Scheduling, Simulated Annealing, Radial Basis Function Network

1. Introduction

Recently, agile and flexible manufacturing has been highly required to deal with diversified customer demands and global competition. Under such circumstances, multi-objective scheduling has been increasingly notified as an important problem-solving in manufacturing. However, since the optimization of scheduling is seriously difficult to solve in itself, its multi-objective optimization has never been studied so much previously.

Among them, Murata, Ishibuchi and Tanaka(5) studied recently about a flow shop problem under two objectives such like makespan and total tardiness using what is known as multi-objective genetic algorithm (MOGA). Bogchi(1) published a book titled "multi-objective scheduling by genetic algorithm". Thereat, flow shop, open shop and job shop problems were concerned under two objectives like makespan and average holding time using a kind of MOGA named non-dominated sorting genetic algorithm (NSGA) and elitist non-dominated sorting genetic algorithm (ENGA). Moreover, Mohri, Masuda and Ishii(4) led a condition existing compromise between makespan and total completion time in flow shop problem. On the other hand, Saym and Karabau(9) used branch and bound method for the similar kind of problem. Parallel machine problem was solved by Tamaki, Nishino and Abe(13) under total holding time and discrepancy from due date using parallel selection-Pareto reserve GA (PPGA). Sakawa and Kubota(8) took job shop problem under three fuzzy objectives by multiple deme GA.

However, these studies stay at deriving only Pareto optimal solution set at most. To work with the problem more extensively, in this paper, we will apply a novel approach of multi-objective optimization named $MOON^{2R}$ ($MOON^2$ of radial basis function), which is derived from $MOON^2$ (Multi-Objective Optimization with value function modeled by Neural Network)(11),(12). It can not only overcome the stiffness and shortcomings of the conventional multi-objective optimization methods, but also derive a best-compro-

* Received 2nd September, 2002 (No. 02-4240).
** Department of Production Systems Engineering, Toyohashi University of Technology, Toyohashi, Aichi 441-8580, Japan. E-mail: shimizu@tutpse.tut.ac.jp

mise solution readily in the unsteady decision environment mentioned already. After giving a general procedure for solving the multi-objective scheduling by $MOON^{2R}$, illustrative examples will be provided to outline the proposed method, and to verify its effectiveness.

2. Soft Computing Approach for Multi-objective Scheduling

2.1 Problem formulation

Generally, we can describe a multi-objective scheduling problem as a multi-objective optimization problem (MOP) described below.

$$(p.1)\ \min \boldsymbol{f}(\boldsymbol{x})=\{f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \cdots, f_N(\boldsymbol{x})\}$$
$$\text{subject to} \quad \boldsymbol{x} \in X$$

where $\boldsymbol{x}$ denotes an n-dimensional decision variable vector, X a feasible region, and $\boldsymbol{f}$ an N-dimensional objective function vector some elements of which conflict and are incommensurable with each other. It should be noted that the above formulation for scheduling refers to integer and/or mixed-integer programming problems[1] whose combinatorial nature makes the solution process very complicated and time consuming (NP-hard).

As a key issue of such MOP, Pareto-optimal solution is popularly known. It provides a rational norm for multi-objective optimization, but never provide a unique or final solution. For any solutions belonging to Pareto-optimal solution set, if we try to improve one objective, we are always urged to degrade another objective as illustrated in Fig. 1. In decision problem, therefore, we have to decide a particular one among a number of solutions by revealing a certain value function of DM explicitly or implicitly (Expressed as a set of contour curves in the figure). Eventually, this means the final solution will be derived through engaging in difficult trade off analysis among the conflicting objectives.

Generally speaking, solution methods of MOP are classified into prior and interactive methods. Each method has advantages and disadvantages over the other. The prior methods try to reveal the preference of decision maker (DM) separately from the searching process. Hence we are not worried about by tedious interactions during the searching process. However, such articulation is inflexible in usual. On the other hand, though interactive methods can articulate elaborately the conflicting objectives, DM must be always ready for the interactions during searching process.

Though the recent studies known as meta-heuristic approach such like MOGA[3],[10] and multi-objective simulated annealing (SA)[2] can deal with the problem in a certain sense, they can generate only the Pareto optimal solution set. As supposed easily, the interactive methods are not available for these algorithms that need the hundreds of interactions. In contrast to it, an approach proposed below ($MOON^{2R}$) can derive a unique solution that should be the best compromise of DM. Hence it becomes a powerful tool for the flexible and agile engineering in real world applications.

2.2 General framework for practical solution

Since $MOON^{2R}$ belongs to a prior articulation method in multi-objective optimization, we need to identify a value function of DM a priori. To improve such modeling stage, we introduced newly a radial-basis function network (RBFN; Orr[6]) instead of usual back propagation network (BPN) employed in $MOON^2$. Due to the linear characteristic of RBFN, the computational load is considerably small compared with BPN. That enables us to model the value function more readily depending on unsteady decision environment popular with scheduling problems. That is, the dynamic adaptation against incremental operation is more flexible and easier than BPN.

The traditional structure of RBFN is shown in Fig. 2. There each component of input vector $\boldsymbol{x}$ feeds forward to the basis functions $\boldsymbol{h}$ whose outputs are linearly combined with the weight $\boldsymbol{w}$ to derive the output $g(\boldsymbol{x})$ as follows:

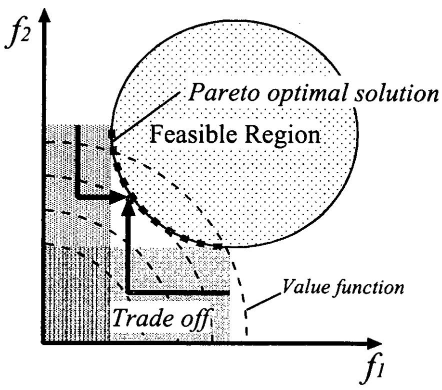

Fig. 1 Idea of solution procedure in MOP

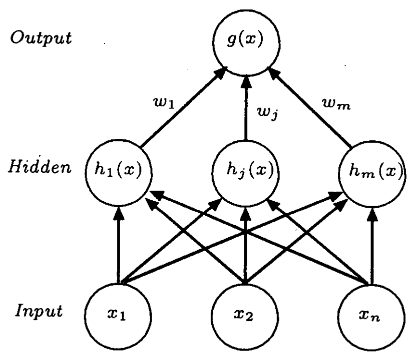

Fig. 2 Traditional structure of RBFN

$$g(\boldsymbol{x})=\sum_{j=1}^{m} w_j h_j(\boldsymbol{x}) \quad (1)$$

Using the training data set such like $(\boldsymbol{x}_i, y_i)$ $(i=1, \cdots, p)$, sum squared error with a penalty term is minimized with respect to the weights (y_i denotes an observed output for input $\boldsymbol{x}_i$).

$$C=\sum_{i=1}^{p}(y_i-g_i(\boldsymbol{x}_i))^2+\sum_{j=1}^{m}\lambda_j w_j^2, \quad (2)$$

where λ_j, $(j=1, \cdots, m)$ denotes regularization parameters.

To train the above RBFN, data regarding the relative preference of DM is gathered through pair comparisons among the appropriate trial solutions. That is, DM is asked to reply which he/she likes, and how much it is between every pair of the trial solutions. Such responses will be taken place by using linguistic statements, and later transformed into the score (Refer to Table 1) like AHP (Analytic Hierarchy Process[7]).

As far as the number of objectives is at most 3, we can express our preference more favorably compared with the pair-comparison taken place in AHP. This is because though we are poor at the comparison between the abstract concepts, e.g., importance between swiftness and cost, we are easy for the comparison between the candidates with concrete contents, e.g., attractiveness between K-rail={swiftness: 2 hrs, cost: 4000 yen} and Jrail={swiftness: 1 hr, cost: 6000 yen}. In fact, this kind of pair-comparison is very popularly encountered in daily decision making.

After doing such pair comparisons over k trial solutions in turn, we can obtain a pair comparison matrix as shown later in Table 2. Its i-j element a_{ij} represents degree of preference (score in Table 1) of $\boldsymbol{f}^j$ compared with $\boldsymbol{f}^i$. According to the same conditions as AHP such that $a_{ii}=1$ and $a_{ji}=1/a_{ij}$, it is necessary for DM to reply only $k(k-1)/2$ times in total. We are also easy to examine the consistency of such pair comparisons from the consistency index calculated by

$$CI=(r_{\max}-k)/(k-1), \quad (3)$$

where $r_{\max}$ denotes the maximum eigenvalue of the pair comparison matrix. It is empirically known if CI value exceeds 0.1, there are involved undue responses in the matrix. In such a case, we need to revise certain scores to recover from the inconsistency.

Table 1 Conversion Table

Linguistic statement	a_{ij}
Equally	1
Moderately	3
Strongly	5
Demonstrably	7
Extremely	9
Intermediate judgments	2,4,6,8

After all, the pair comparison matrix provides totally k^2 training data for RBFN as shown already. Every objective value of the pair, say, $\boldsymbol{f}^i$ and $\boldsymbol{f}^j$ becomes $2N$ inputs, and i-j element a_{ij} an output of RBFN. We are possible to view thus trained RBFN as an implicit function, or a mapping from a $2N$ dimensional space Φ to a scalar, i.e. V_{RBF}: $(\boldsymbol{f}^i(\boldsymbol{x}), \boldsymbol{f}^j(\boldsymbol{x}))\in\Phi\subset R^{2N}\rightarrow a_{ij}\in R^1$. Here, notice that the following relation will hold.

For a certain pair of $\boldsymbol{f}^i, \boldsymbol{f}^j, \boldsymbol{f}^k$

$$V_{RBF}(\boldsymbol{f}^i, \boldsymbol{f}^k)=a_{ik}\geq V_{RBF}(\boldsymbol{f}^j, \boldsymbol{f}^k)=a_{jk} \Leftrightarrow \boldsymbol{f}^i \gtrsim \boldsymbol{f}^j \quad (4)$$

Now we can rank the preference of any solutions easily by the output of RBFN, a_{*R} calculated by fixing one of the input vector at an appropriate reference, say $\boldsymbol{f}^R$:

$$V_{RBF}(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}^R)=a_{*R} \quad (5)$$

That is, trajectories with the same values of R.H.S. of Eq.(5) provide the indifference curves or contours of value function imposed in Fig. 1. Such an ranking is valid as long as the inconsistency of the pair comparison is satisfied (i.e., $CI<0.1$).

After all, the foregoing MOP (p. 1) is possible to describe as follows.

(p. 2) max $V_{RBF}(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}^R)$ subject to $\boldsymbol{x}\in X$

The above formulation will be supported by the following proposition.

[**Proposition**] The optimal solution of (p. 2) derives a Pareto optimal solution of (p. 1) if value function is identified so as to satisfy the relation given by Eq(4).

(Proof) Let f_i^*, $(i=1, \cdots, N)$ be each value of objective function for the optimal solution of (p. 2), $\boldsymbol{x}^*$, i.e., $f_i^*=f_i(\boldsymbol{x}^*)$ (Hereinafter, discuss in the objective function space.) First assume $\boldsymbol{f}^*$ is not a Pareto optimal solution. Then there exists a certain $\boldsymbol{f}^0$ such that for $\exists j, f_j^0<f_j^*-\Delta f_j, (\Delta f_j>0)$ and $f_i^0\leq f_i^*, (i=1, \cdots, N, i\neq j)$. Since DM apparently prefers $\boldsymbol{f}^0$ to $\boldsymbol{f}^*$, it holds that $V_{RBF}(\boldsymbol{f}^0, \boldsymbol{f}^R)>V_{RBF}(\boldsymbol{f}^*, \boldsymbol{f}^R)$. This contradicts that $\boldsymbol{f}^*$ is the optimal solution of (p. 2). Hence $\boldsymbol{f}^*$ must be a Pareto optimal solution.

Thus describing the multiple objectives into an overall one, we can apply a variety of optimization methods known previously, i.e., nonlinear programs, direct search methods, and even more meta-heuristic methods like GA, SA, Tabu search, etc. Among them, SA is considered favorable due to certain combinatorial natures of scheduling problems. Its application is straightforward since we can evaluate any candidates under the multi-objectives through V_{RBF} once $\boldsymbol{x}$ is given.

SA is viewed as a randomized neighborhood search algorithm. It uses an analogy with the physical process of annealing, in which a pure lattice structure of a solid is made by heating up the solid in a heat bath until it melts, then cooling it down slowly until it solidifies into a low energy stage. During the iteration, a neighborhood solution is generated randomly around the current solution, and it is checked if acceptable or not. Only if acceptable, the current solution is revised, and the same procedure will be repeated until a certain convergence criterion has been satisfied. It is said, due to the stochastic perturbation, the algorithm can hopefully attain at the optimal point without trapping into local solutions.

After all, we can summarize the proposed solution method as follows (Refer also Fig. 3).

1. Generate several trial solutions in objective function space.
2. Ask the preference of DM through pair comparison between every pair of the trial solutions.
3. Train RBF based on the above result. This provide a value function V_{RBF}.
4. Finally, apply SA to solve the problem (p. 2).
5. If DM is unsatisfactory with the result, limit the searching space around there, and repeat the same procedure untill he/she likes the result.

In the above approach, since the modeling process of the value function is separated from the searching process, DM can carry out his/her tradeoff analyses at his/her own pace without worrying about the hurried/idle responses often experienced at the interactive MOP methods. In addition, since the required responses are simple and relative, DM's load in such interaction is very small. Moreover, modeling by RBFN can deal adaptively with the change of the decision environment that makes likely alter the preference of DM. Even in such a case, its retraining is easily taken place through incremental operations against both increase and decrease in the training data and bases from the foregoing one as will be discussed briefly in the latter. These are particular advantages for aiming at the agile and flexible decision making.

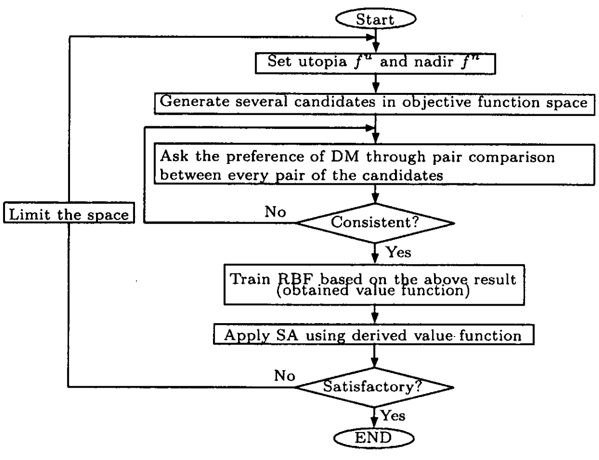

Fig. 3 Flow chart of the proposed solution procedure

3. Illustrative Example

To examine the effectiveness of the proposed approach, we solved multi-objective flow shop scheduling problems under two objective functions i.e. minimization of sum due time delay f_1 and total changeover cost f_2. We also imposed mild assumptions such that any jobs are not dividable, simultaneous operation, are inhibited, and processing time and due date are given.

As a generic property of MOP (subjective decision problem), it is impossible to derive a preferentially optimal solution only by the mathematically provided conditions. Hence to verify the effectiveness of the method in the numerical experiments, we supposed the virtual DM whose preference is given as a value function defined by

$$U(\boldsymbol{f}(\boldsymbol{x}))=\left[\sum_{i=1}^{N} w_i\left\{\frac{f_i(\boldsymbol{x})-f_i^{nad}}{f_i^{utop}-f_i^{nad}}\right\}^p\right]^{1/p}, \quad (p=1, 2, \cdots) \qquad (6)$$

where w_i is a weight factor and p a parameter to specify the adopted norm respectively. On the other hand, f_i^{utop} and f_i^{nad} denote utopia and nadir values respectively.

Moreover, we need to characterize the virtual DMs more minutely to simulate their preference i.e., subjective judgment in their pair comparisons. That is, the degree of preference mentioned already in Table 1 is assumed to be given as

$$\begin{cases} a_{ij}=1+\left[\dfrac{8(U(\boldsymbol{f}^i)-U(\boldsymbol{f}^j))}{U(\boldsymbol{f}^{utop})-U(\boldsymbol{f}^{nad})}+0.5\right] \\ \qquad\qquad \text{if } U(\boldsymbol{f}^i)>U(\boldsymbol{f}^j) \\ a_{ij}=1/a_{ji} \qquad \text{Otherwise} \end{cases} \qquad (7)$$

where [•] denotes Gauss's operator. Then to verify the effectiveness, we compared the result obtained from the proposed approach to the reference solution that is derived from the direct optimization under Eq. (6).

Among the trial solutions generated as shown in Fig. 4, the pair comparison matrix of the virtual DM is given as Table 2 based on Eq. (7) ($p=1$, $w_1=0.3$, $w_2=0.7$). Since only the upper triangular part should be prepared, total number of such responses becomes 35 in this case ($a_{utop \cdot nad}=9$ is implied). Using the

Table 2 Pair comparison matrix ($p=1$)

	F^u	F^n	F^1	F^2	F^3	F^4	F^5	F^6	F^7
F^u	1	9	3	7	5	3	7	4	6
F^n	1/9	1	1/7	1/3	1/5	1/7	1/3	1/6	1/4
F^1	1/3	7	1	4	3	1	4	2	3
F^2	1/7	3	1/4	1	1/3	1/4	1	1/3	1
F^3	1/5	5	1/3	3	1	1/3	3	1/2	2
F^4	1/3	7	1	4	3	1	5	2	4
F^5	1/7	3	1/4	1	1/3	1/5	1	1/4	1/2
F^6	1/4	6	1/2	3	2	1/2	4	1	3
F^7	1/6	4	1/3	2	1/2	1/4	2	1/3	1

normalized values of those, we trained the RBFN to obtain the value function defined in Eq.(5).

We compared the contour lines of preference (indifference curves) between the supposed and $V_{RBF}(\boldsymbol{f}, \boldsymbol{f}^R)$ in Fig. 5 when $p=1$. Except for the marginal regions, we confirmed that RBFN could model the supposed value function correctly*[1].

Under thus identified value function, we solved three flow shop scheduling problems i.e.,

1. one process, one machine and 7 jobs
2. two processes, one machine and 10 jobs
3. two processes, two machines and 10 jobs.

Each objective function was specified by generating randomly the scheduling data within certain extents i.e., between 1 and 10 for f_1 and every 4 interval between 4 and 40 for f_2 respectively.

Regarding SA applied here as an optimization method, we adopted the insertion neighborhood method, and gave tuning parameters such that reduction rate of temperature=0.95, and number of iteration=400.

In Table 3, we summarized some numerical results in comparison with the reference solutions, and

Table 3 Comparison of numerical results ($p=1$)

	Overall objective func.	
*Kind of problem**	Reference	V_{RBF}
(1,1, 7)	3.47	3.47
(2,1,10)	7.95	7.95
(2,2,10)	3.40	3.40

*Numbers of (process:S, machine:M, job:J)

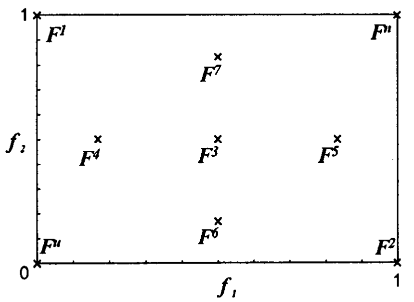

Fig. 4 Location of trial solutions

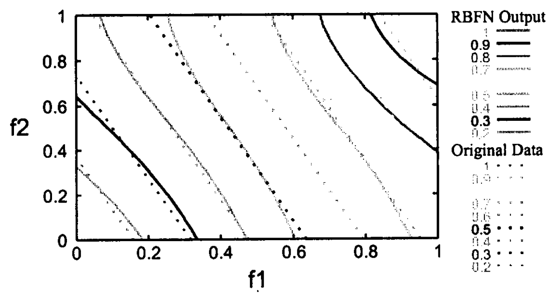

Fig. 5 Comparison of contour of value function ($p=1$)

*1 Presently $\boldsymbol{f}^R$ is set at (0, 0)

provide a Gantt chart in Fig. 6 to give a visual examination of the feasibility of the result. Same results in every case ascertain that the proposed method can solve the problem correctly through accurate identification of the virtual DM's preference by V_{RBF}.

Moreover, to examine the applicability in the real-life situation, we decreased the number of trial solutions gradually from the foregoing 9 to 7 and 5. Accordingly number of required replies will be decreased by 20 and 10 respectively. Those numbers are small enough for DM to respond acceptably. Also in every case, we could obtain the same results as shown in Table 3. This means we can identify the linear value function correctly with small load of interaction.

Just like the same way, we solved the problems successfully for the case of quadratic form of value function as shown both in Fig. 7 and Table 4. From these results, we can ascertain that our approach is possible to cope with the various types of value function.

For further consideration, we should notice the following points. If we failed to model the value function properly due to the complicated nonlinearity, we need to refine it around the first solution until satisfactory result will be obtained (This happened in

Table 4 Comparison of numerical results ($p=2$)

	Overall objective func.	
Kind of problem	Reference	V_{RBF}
(1,1, 7)	1.40	1.40
(2,1,10)	2.92	2.92
(2,2,10)	1.60	1.60

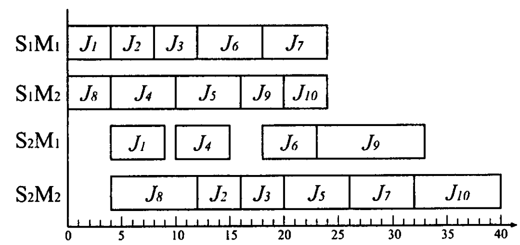

Fig. 6 Gantt chart of (2, 2, 10) problem ($p=1$)

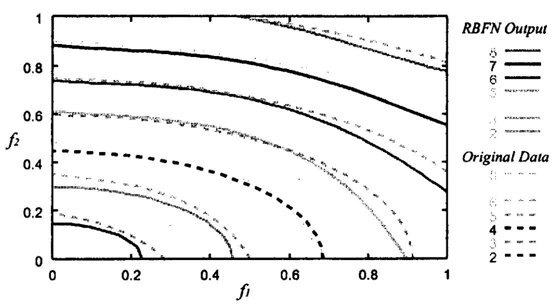

Fig. 7 Comparison of contour of value function ($p=2$)

Table 5 Load of incremental operation[5]

Operation	Completely retrain	Use operation
Add/Remove a basis	$m^3+pm^2+p^2m$	p^2
Add/Remove a pattern	$m^3+pm^2+p^2m$	$2m^2+pm+p^2$

the Min-max ($p=\infty$) case). Such situation happens also in the case where we need to modify the value function against moving and/or changed preferences. To deal with these problems properly, it is necessary to have a procedure easy for incremental operations in the network modeling. Regarding this point, RBFN has a nice property. Noticing the relations in Eqs.(8) and (9), for example, revised calculation is given by Eq.(10) in the case of adding a new training pattern.

$$\boldsymbol{A}_p=\boldsymbol{H}_p^{\top}\boldsymbol{H}_p+\Lambda \tag{8}$$

$$\boldsymbol{H}_{p+1}=\begin{bmatrix}\boldsymbol{H}_p\\ \boldsymbol{h}_{p+1}^{\top}\end{bmatrix} \tag{9}$$

$$\boldsymbol{A}_{p+1}^{-1}=\boldsymbol{A}_p^{-1}-\frac{\boldsymbol{A}_p^{-1}\boldsymbol{h}_{p+1}\boldsymbol{h}_{p+1}^{\top}\boldsymbol{A}_p^{-1}}{1+\boldsymbol{h}_{p+1}^{\top}\boldsymbol{A}_p^{-1}\boldsymbol{h}_{p+1}} \tag{10}$$

where $\boldsymbol{A}_p$ is a variance matrix, and $\boldsymbol{H}_p=(\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_p)$ denotes a design matrix. On the other hand, when removing an old training pattern, we use the relation in Eq.(11).

$$\boldsymbol{A}_{p-1}^{-1}=\boldsymbol{A}_p^{-1}+\frac{\boldsymbol{A}_p^{-1}\boldsymbol{h}_i\boldsymbol{h}_i^{\top}\boldsymbol{A}_p^{-1}}{1+\boldsymbol{h}_i^{\top}\boldsymbol{A}_p^{-1}\boldsymbol{h}_i} \tag{11}$$

Load required for retraining by these incremental operations is roughly estimated as shown in Table 5. As the number becomes large, the effect of time-saving is known to become considerably large.

4. Conclusion

To solve the multi-objective scheduling problem, in this paper, we have proposed a practical approach characterized, within the framework of *MOON*^{2}, by the modeling process of the value function by RBF and the adaptation of SA as a optimizer. By thus developed approach named *MOON*^{2R}, we can deal with the unsteady decision environment popular with the real-life scheduling problems by virtue of its flexible and simple practice. Illustrative examples are provided to outline the proposed method, and verify its effectiveness.

References

(1) Bagchi, T.P., Multi-objective Scheduling by Genetic Algorithms, (1999), Kluwer Academic Publishers, USA.

(2) Czyzak, P. and Jaszkiewicz, A. Pareto Simulated Annealing—A Meta-heuristic Technique for Multiple Objective Combinatorial Optimization, J. Multi-criteria Decision Analysis, Vol.7 (1998), pp.34-47.

(3) Fonseca, C.M. and Fleming, P.J., Genetic Algorithm for Multi-Objective Optimization: Formulation, Discussion and Generalization, Proc. 5th Int. Conf. on Genetic Algorithm and Their Applications, Chicago, (1993), pp.416-423.

(4) Mohri, S., Masuda, T. and Ishii, H., Bi-criteria Scheduling Problem on Three Identical Parallel Machines, Int. J. Production Economics, Vol.60-61 (1999), pp.529-536.

(5) Murata, T., Ishibuchi, H. and Tanaka, H., Multi-Objective Genetic Algorithm and Its Applications to Flowshop Scheduling, Computers Ind. Engng, Vol.30, No.4 (1996), pp.57-968.

(6) Orr, M.J.L., Introduction to Radial Basis Function Networks,, http://www.cns.uk/people/mark.html (1996).

(7) Saaty, T.L., The Analytic Hierarchy Process, (1980), McGraw-Hill, New York.

(8) Sakawa, M. and Kubota, R., Fuzzy Programming for Multi-objective Job Shop Scheduling with Fuzzy Processing Time and Fuzzy Due date through Genetic Algorithms, Europe. J. Oper. Res., Vol.120 (2000), pp.393-407.

(9) Saym, S. and Karabau, S., Bicriteria Approach to the Two-Machine Flow Shop Scheduling Problem, Europe. J. Oper. Res., Vol.113 (1999), pp.435-449.

(10) Schaffer, J.D., Multiple Objective Optimization with Vector Evaluated Genetic Algorithm, Proc. 1st Int. Conf. on Genetic Algorithm and Their Applications, Pittsburgh, (1985), pp.93-100.

(11) Shimizu, Y., Multi-objective Optimization for Site Location Problems through Hybrid Genetic Algorithm with Neural Networks, J. Chem. Engng, Japan, Vol.32 (1999), pp.51-58.

(12) Shimizu, Y. and Kawada, A., Multi-Objective Optimization in Terms of Soft Computing, Trans. of SICE, Vol.38 (2002), pp.974-980.

(13) Tamaki, H., Nishino, E. and Abe, S., Modeling and Genetic Solution for Scheduling Problems with Regular and Non-Regular Objective Functions, Trans. SICE, (in Japanese), Vol.35, No.5 (1999), pp.662-667.