An approach to design of maintenance float systems

Mu-Chen Chen
Department of Business Management,
Institute of Commerce Automation and Management,
National Taipei University of Technology, Taipei, Taiwan, ROC
Hsien-Yu Tseng
Department of Industrial Engineering and Management,
St John's and St Mary's Institute of Technology, Tamsui, Taiwan, ROC

Keywords
Maintenance, Simulation, Modelling, Neural networks, Genetic algorithms

Abstract
The paper offers an intelligent approach to analyze and determine the design parameters minimizing the total cost and achieving the desired performance measures in the maintenance float systems. The expected total cost in a maintenance float system includes the cost of lost production, the cost of repair persons and the cost of standby machines. The developed design procedure integrates simulation, metamodel and genetic algorithms. Neural networks are able to approximate functions based on a set of sample data, i.e. construct metamodels from simulation results in this study. The objective of metamodels is to predict simulation responses in order to significantly reduce the amount of simulation runs. The predictive performance of neural metamodels comparably outperforms the traditional regression metamodels. The neural metamodels are further extended to formulate a decision model for optimizing the maintenance float systems by using genetic algorithms.

Received May 2001
Revised February 2002
Accepted May 2002

Integrated Manufacturing Systems
14/5 [2003] 458-467
© MCB UP Limited
[ISSN 0957-6061]
[DOI 10.1108/09576060310477861]

1. Introduction

Owing to the increasing dependence on advanced manufacturing technologies, the equipment availability and reliability become a critical concern in maintenance. The effective maintenance float system is constructed to minimize the cost of lost production incurred by the inherent unreliability in machines. The information of equipment availability and reliability will facilitate in the planning and allocation of resources to perform necessary tasks and services.

A maintenance float system can be considered as a closed queueing network (refer to Figure 1), in which the machines alternate among operation, repair and standby status (Madu and Chanin, 1992; Madu and Kuei, 1996). In a maintenance float system (Madu and Chanin, 1992), N independent and identical machines are required to be in operation at the same time. When a failure occurs, the number of operating units is reduced to $N-1$. If there exists any standby unit, the failed unit is replaced to restore the number of operating units to N. The failed unit goes in for repair and is returned to a standby status if the total number of operating units at the time of completing repair is up to N. Otherwise, it is sent into operation. It is assumed that the failed unit is completely rejuvenated after repair.

The expected total cost $\overline{TC}$ in a maintenance float system includes the cost of lost production, the cost of repair persons and the cost of standby machines. The cost of lost production is incurred when a machine breaks down and waits for repair. The expected total cost can be stated as (Madu and Chanin, 1992):

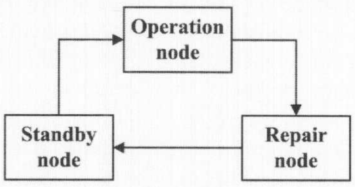
$$TC = \sum_{N \geq F}^{N} C_1(n-F)P(n) + C_2S + C_3F, \quad (1)$$

where N is the number of independent and identical machines initially in operation or required in operation at all times; S is the number of independent and identical repair persons; F is the maintenance float required to support the N machines that are in operation or standby units; C_1 is the cost of equipment downtime; C_2 is the cost per repair person; C_3 is the cost of maintaining a standby unit, and $P(n)$ is the probability that n units are down.

In order to design an effective maintenance float system, a model to represent the relationships between various important factors and system performance is required. The intrinsic complexity in the maintenance system makes analytical models difficult for presenting the actual environment. Effective analytical approaches for large-scale systems with non-Markovian failure distributions are generally hard to obtain (Madu and Kuei, 1996). Even for small-scale systems with Markovian properties, the existing procedures require complicated algorithms and numerous iterations (Gross *et al.*, 1983; Mani and Sarma, 1984; Madu, 1988a, b; Madu *et al.*, 1990). Approximate models and efficient approaches are therefore necessary to address the maintenance float systems.

The modeling and design of a real-world production system is a difficult and complicated mission. Simulation is frequently utilized to study complex production systems that cannot be resolved by analytical methods (Szczerbicki, 2000). Disregarding analytical models, simulation models have a widespread acceptance in studying maintenance float systems (Chanin *et al.*, 1990; Lin, 1996; Madu *et al.*, 1990; Sahu and Sharam, 1984). Furthermore, Chanin

Figure 1
The maintenance float network



et al. (1990), Lin (1996), Madu and Chanin (1992) and Madu and Kuei (1996) applied hybrid approaches, which combine simulation and regression metamodeling to study the maintenance float problems.

It has generally been complicated to analyze maintenance float networks by analytical models. Computer simulation (Madu and Chanin, 1992) offers a considerably more flexible modeling approach to address various maintenance float problems (e.g. complex failure distributions such as gamma and Erlang-2). However, simulation commonly generates a lot of data and tables, and they are difficult to interpret without statistics summary. Metamodels can be applied to release this major limitation of simulation technique.

In the case of regression metamodeling, one must make assumptions about the form of the regression equation, which may not be straight to model builders. Neural networks recently have been reported to be a successful statistics tool for approximating the relationships between input factors and output responses. In contrast, neural networks attempt to map relationships through data without giving a predetermined function with free parameters. Neural networks therefore can identify unfamiliar functional relationships, which are unknown to managers, between controllable factors and performance measures in the maintenance float networks.

This study aims to develop a simulation metamodeling procedure based on neural networks, namely neural metamodeling method. The neural metamodels are further applied to formulate the decision model for optimizing the maintenance float system. An optimization method based on genetic algorithms (GAs) is developed to resolve the mathematical model for the economic design of maintenance float systems.

2. The design approach

This study proposes a hybrid analysis approach, which incorporates simulation, neural metamodel and genetic algorithms (GAs) to address the maintenance float problems. The neural network is adopted to construct the metamodels of the simulated systems (estimate the relationships between the design variables and simulation responses). A GA-based optimization method is finally applied to obtain the optimal combination of design variables for the simulated maintenance problem. A single simulation analysis of large-scale systems requires a huge amount of execution time and memory. Therefore, it is necessary to develop a vigorous approximation to minimize the number of simulation runs during the optimal systems design process. The proposed intelligent approach for the economic design of maintenance float systems can reduce CPU time and save computer memory significantly.

The proposed design procedure for maintenance float systems has the following seven steps:

1 *Define the maintenance float problem*. For economic design of manitenance float systems, several primary points should be issued (Lin, 1996):
   - What is the maintenance problem of the system?
   - Which performance measures are to be used?
   - Which factors are involved in the system?
   - Which factors are the independent and dependent variables?

2 *Construct the simulation model*. The simulation model can then be constructed through a detailed consideration of the above issues, and the factors in the experimental design can be selected. This model is applied for examination in the design of a maintenance float system. Simulation can be adopted to generate the appropriate data for analysis of the system performance. Simulation modeling has been widely discussed in simulation textbooks (e.g. Law and Kelton, 1991).

3 *Build the experimental design*. In the experimental design for simulation, the input parameters and structural assumptions composing a model are called factors, and the output performance measures are called responses. A simulation study usually consists of several performance measures or responses of interest. In this step, designers identify the controllable factors which might affect the performance measures (simulation responses) of interest. The fractional design or Taguchi design for experimental study can

delimitate the quantity of data collection (Kuei and Madu, 1996).

4 *Run the simulation.* In simulation, experimental runs will be conducted with various settings of the controllable factors (design variables) according to carefully designed experiments. Owing to the random nature of a simulation model, a single simulation run for each design point is insufficient to describe the simulation results. Various random number seeds are applied to simulate each design point, and the average simulation results are taken as the output of each design point.

5 *Construct the neural metamodel.* The simulation results are collected for training the neural networks as metamodels. Another set of simulation runs is taken as validation data to test the accuracy of trained neural metamodels. The influencing factors are fed into the input nodes of the neural network, and the responses of interest are target values associated with output nodes. As the training procedure begins, the input-target patterns from simulation results are presented to the networks. The error of approximation is applied to measure the performance of neural networks. Once the obtained error (e.g. mean squared error, MSE) is less than the specified error level, the neural metamodels can be generated. The next section provides the detailed neural metamodeling technique.

6. *Formulate the decision model.* The nonlinear functions (neural metamodels), which depict the approximation between the design variables and simulation responses, are generated from the trained neural networks. The decision model can be formulated for economic design of maintenance systems according to the primary objective of systems design. For the mathematical model, refer to Section 4.

7 *Solve the decision model.* The functions represented by the trained neural metamodels, which represent the relationships between the controllable factors and simulation responses, are generally nonlinear functions for the original data set. A GA-based optimization method is adopted to optimize the nonlinear decision models stemming from the neural metamodels. Section 5 will discuss the further details of GA-based optimization method.

3. Neural metamodeling

Simulation is generally time and memory consuming. Additionally, the simulation by itself cannot generate an optimal solution to the simulated problem. Metamodels, models of simulation models, are built to easily interpret the simulated systems and to allow some explorations from the simulated range of system states. By using the metamodeling technique, the simulation model can be simplified and the model execution time is reduced and memory is saved dramatically (Law and Kelton, 1991).

Regression metamodeling was first proposed by Blanning (1975). The applications of metamodeling in simulation have been extensively developed by Kleijnen (1987) and Law and Kelton (1991). Apart from approximation approaches, the neural metamodel (approximation via neural network) is similar to regression metamodel. Let x_j denote a j factor influencing the outputs of the real-world system $(j = 1, 2, \dots, r)$. And, let $\mathbf{Y}$ denote the system response vector $(\mathbf{Y} = \{y_k | k = 1, 2, \dots, n\})$. We can concentrate on a system with a single response y_k to simplify the discussion. It will not lose the generality since a multi-response system can be viewed as a set of single response systems. The metamodel of each response in the multi-response system can be obtained by using a similar process. The response variable y_k is the functions of the factor vector $\mathbf{X}$. It takes the form as:

$$y_k = f_1(x_1, x_2, \dots, x_r). \quad (2)$$

A simulation model is an abstraction of the real world system, in which we consider only a subset of the input factors $\{x_j | j = 1, 2, \dots, s\}$. Generally, s is significantly smaller than the unknown r. The simulation response y'_k is then defined as a function f_2 of this subset and a vector of random numbers v representing the effect of the excluded inputs. In this case, y'_k is a function f_2 of s factors, plus a vector of random numbers v. The relationship can be expressed as:

$$y'_k = f_2(x_1, x_2, \dots, x_s, v). \quad (3)$$

A neural metamodel is then a further abstraction, where we select a subset of the simulation input variables $\{x_j | j = 1, 2, \dots, m, \ m \leq s\}$ and depict the system as:

$$y''_k = f_3(x_1, x_2, \dots, x_m) + \epsilon, \quad (4)$$

where ϵ denotes a fitting error having an expected value of zero. The simulation model represented by f_3 may be approximated in turn by a neural metamodel within a specific experimental range. The above abstraction for neural metamodel is depicted in Figure 2.

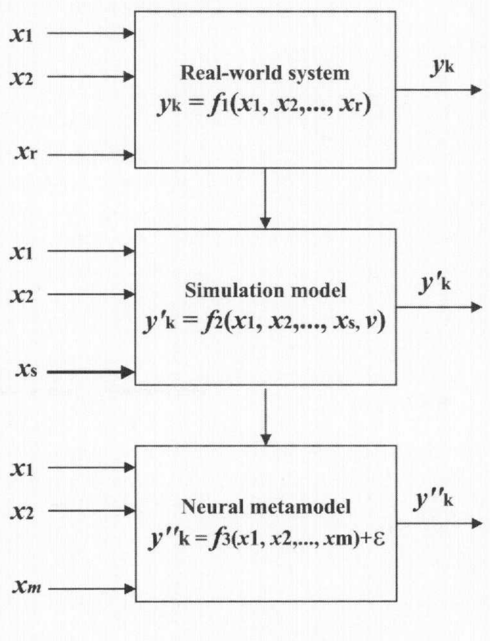
One of the major benefits of neural networks is the adaptive ability of generalization of data from the real world

(Lippmann, 1987). Taking this advantage, many researchers apply neural networks for nonlinear regression analysis and have enhanced results compared to regression in their applications (e.g. Bode, 2000; Chen, 2001). Recently, neural networks have received a great deal of attention in production areas. Zhang and Huang (1995) presented a rather extensive review of neural network applications in production.

The proposed approach towards the simulation metamodeling problem is based on supervised neural networks. Backpropagation (BP) neural network is most extensively used and can provide good solutions for many industrial applications (Lippmann, 1987). Further, from experience and literature reviews, it has been found that the three-layer BP network is suitable for almost all problems if enough hidden neurons are used. Neural networks are expected to demonstrate comparative advantages to regression in simulation metamodeling.

In this paper, neural networks are used to fit the form of f_3 to determine the relationship of design variables, $\{x_j | j = 1, 2, \ldots, m, m \leq s\}$, and interesting simulation responses (performance measures) which are associated with the primary objective of systems design. The neural network approach can be regarded as a statistical method. Neural networks can learn the relationship between design parameters and responses from observed data. However, the data for teaching neural networks need to be cautiously selected. In this paper, statistical design of experiments is used to observe the relationship information of interest from the minimum number of simulation runs.

Figure 2
The neural metamodeling


x_1
x_2
x_r
Real-world system
$y_k = f_1(x_1, x_2, \ldots, x_r)$
y_k
x_1
x_2
x_s
Simulation model
$y'_k = f_2(x_1, x_2, \ldots, x_s, v)$
y'_k
x_1
x_2
x_m
Neural metamodel
$y''_k = f_3(x_1, x_2, \ldots, x_m) + \varepsilon$
y''_k

Experimental designs such as full factorial design, fractional factorial design and Taguchi design are often employed to determine the values of input variables (Kleijnen, 1987). Provided that the data are selected using the orthogonal array, the accurate result using smaller observed database might be acquired (Montgomery, 2000). The factorial design in high dimensional problems requires a huge set of experimental data. The fractional factorial design and Taguchi method need a smaller amount of simulation to obtain the desired output information, particularly in high dimensional problems. For experimental designs, two level designs (2^k) are usually adopted due to its simplicity. However, it is only an appropriate estimator for linear relationships. Although higher order designs (e.g. 3^k) make the experimental design more complicated, it is useful under the situation of nonlinear relationships between input factors and output responses. The designers should select the suitable experimental design for metamodeling with respect to several aspects involving the number of input factors, experimental cost and data analysis accuracy.

Since this study aims to optimize the maintenance float system with minimal simulation runs, an orthogonal array $L_{27}(3^{13})$ is adopted to systematically and cautiously obtain a smaller amount of simulation than that of factorial design. In metamodeling, the orthogonal array is used to limit the training samples. This saves time and reduces the cost of conducting a large number of simulation experiments. However, it may induce the over-fitting effect. In over-fitting, the neural network's output fits the training data too closely (Smith, 1993). It models the noise in addition to the underlying function we want it to find. We can restrict the over-fitting by limiting the hidden nodes in the network and by limiting the training epochs. BP as well as other modeling approaches would achieve a higher level of accuracy if more data were provided. However, BP neural network does not require a larger sample than regression (Smith, 1993) to achieve the similar accuracy.

In addition, the data used for training a neural metamodel in this study has been obtained from a simulation model. Each systems design has been replicated a number of times and so the data presented to neural networks is already smooth and with lower noise. The systems designer may control the accuracy of the results for each design by running more replications and so the noise component in the simulation data can be reduced. Therefore, the data is more desirable for the effective training of a neural metamodel.

4. The decision model

For metamodels, four general goals are identified by Kleijnen and Sargent (2000):

1 understanding the problem entity;
2 predicting values of the output or response variable;
3 performing optimization; and
4 aiding the verification and validation of a simulation model.

This study primarily focuses the post-metamodeling analysis on simulation optimization.

In this paper, the objective is to determine the number of machines initially in operation (*N*), the number of standby machines (*F*), the number of repair persons (*S*) and the appropriate expected repair time (*R*) to minimize the expected total cost (*TC*), depicted in equation (1). Meanwhile, some performance measures of maintenance float systems such as average equipment utilization (*AEU*), average repair person utilization (*ASU*) and expected waiting time for repair (*AWT*) are also intended to satisfy. The manager may be interested in keeping the equipment's availability higher than a load level. While the manager wants to effectively manage the repair persons, their workload may not be overloaded. At the same time, the waiting time for repair should be within a reasonable range.

The decision model for the maintenance float problem can be formulated as:
Minimize:

$$TC = \sum_{n \geq F}^{N} C_1(n - F)P(n) + C_2S + C_3F$$

$$= C_1 \times AEU \times N + C_2S + C_3F.$$

Subject to:

$$AEU \geq \theta_1; \quad (5)$$

$$ASU \leq \theta_2;$$

$$AWT \geq \theta_3 \text{ hr};$$

$$AWT \leq \theta_4 \text{ hr}$$

where θ_1 is the lower limit of *AEU*; θ_2 is the upper limit of *ASU*; and (θ_3, θ_4) is the allowed range for *AWT*. In the above decision model, *AEU*, *ASU* and *AWT* are functions in terms of *N*, *S*, *F* and *R*. These functions are obtained from the neural metamodels.

5. Optimal design using genetic algorithms

Genetic algorithm (GA), developed by Holland (1975) is a part of evolutionary algorithms, which is a rapidly growing area of optimization. GA is found to be useful where the search space is large, nonlinear and noisy, and solutions are ill defined a priori (Fogel, 1994; Goldberg, 1989). Recently, various ideas for combining GAs and neural networks have been proposed and investigated (Sarkar and Yegnanarayana, 1998). There characteristically exist two sorts of combinations (Schaffer *et al.*, 1992):

1 supportive combinations (they are used sequentially); and
2 corporative combinations (they are used simultaneously).

Supportive combinations typically involve using one of these methods to prepare data for use by the other. For instance, GA is used to select features for neural network classification in the stage of data pre-processing. Corporative combinations typically contain the use of GAs to determine the network connection weights or network architectures or both. From the survey by Sarkar and Yegnanarayana (1998), GAs are successfully used to optimally configure neural networks so that they have better learning capability.

This study proposes a supportive combination of GAs and neural networks to simulation analysis for optimum systems design. However, the proposed systems design approach incorporates GAs with neural networks for post-training process. Neural networks are applied to fit the behavior of design systems (generate simulation metamodels), and GAs are then used to optimize the design systems based on the trained neural networks. The distinctive benefit of the developed approach is the extensive reduction of simulation runs and computational requirement required to generate an optimum design.

For a complete discussion of GA and its applications, refer to Goldberg (1989). The general schema of a GA is as follows:

- *Step 1.* Start with a population of individuals.
- *Step 2.* Evaluate fitness of all initial individuals.
- *Step 3.* Select subpopulation (parents) stochastically for children reproduction (selection operator).
- *Step 4.* Recombine selected parents stochastically (crossover operator).
- *Step 5.* Perturb the mated population stochastically (mutation operator).

- *Step 6*. Evaluate the fitness of mated population.
- *Step 7*. Check for termination criterion and stop or return to Step 3.

Several basic items of the specific GA implementation for economic design of maintenance float systems are discussed as follows:

- *Encoding*. A real value encoding is used herein, so that each solution of design parameters is encoded as a vector of real value coefficients. For the integer variable, it is obtained from rounding the continuous value to the nearest integer.
- *Evaluation*. The real values in X are then inserted into the mathematical model (refer to Section 4) to obtain the relative objective function value. For roulette wheel selection method, it is required to transform the objective values into fitness values in such way that the fitter one has the larger fitness values.
- *Arithmetic crossover*. The crossover operator applied here is arithmetic crossover (Gen and Cheng, 1997). The arithmetic crossover is defined as the combination of two chromosomes $\mathbf{X}_1$ and $\mathbf{X}_2$ as follows:
$$\mathbf{X}_1 = r\mathbf{X}_1 + (1-r)\mathbf{X}_2;$$
$$\mathbf{X}_2 = r\mathbf{X}_2 + (1-r)\mathbf{X}_1, \qquad (6)$$
where $r \in (0,1)$. The probability of crossover is set as p_c, i.e. on average, $p_c \times 100\%$ of chromosomes undergo crossover.
- *Nonuniform mutation*. The nonuniform mutation (Gen and Cheng, 1997) is utilized in this algorithm. For a given parent $\mathbf{X}$, if the element x_k of it is selected for mutation, the resulting offspring is $[x_1, \ldots, x'_k, \ldots, x_n]$, where x'_k is randomly selected from the two possible choices (with equal probability):
$$x'_k = x_k + \Delta(t, x_k^U - x_k)$$
or
$$x'_k = x_k - \Delta(t, x_k - x_k^L), \qquad (7)$$
where x_k^U and x_k^L are the upper and lower bounds for x_k. The function $\Delta(t,q)$ returns a value in the range in [0, q] such that the value of $\Delta(a,q)$ approaches to zero as t increases (t is number of current generation). It takes the form as:
$$\Delta(t,q) = q \times r \times \left(1 - \frac{t}{T}\right)^b, \qquad (8)$$
where r is a random number from [0, 1], T is the maximum number of generations, and b is a parameter determining the degree of nonuniformity. The probability of mutation is set as p_m, i.e. on average, $p_m \times 100$ percent of total elements in population would undergo mutation. Every element has an equal chance to be mutated.

6. A maintenance float system

6.1 Description of the problem

The maintenance float system studied herein is based on the illustrative example presented by Madu and Kuei (1996). This example considers a manufacturing system that operates on a 24-hour operation with several independent and identical machine tools. Figure 1 presents this maintenance float problem. Both the failure and repair times of these machines follow the Erlang-2 distribution. When a machine tool fails, a standby unit replaces it, if any exists, while it goes through repair. The operations manager is interested in satisfying the equipment's availability load 90 per cent of the time. Meanwhile, the manager wants the expected waiting time for repair to be within [0.5, 1.0] hour. To effectively manage repair persons, the utilization of repair persons is expected to be higher than 0.85.

To achieve the above performance measures, the manager would like to know:

- the number of machine tools required;
- the number of standby machine tools;
- the number of repair persons; and
- the expected repair time.

The manager makes a simulation experiment to answer the above questions. These design variables are taken as controllable factors in the simulation experimental design. The output information of simulation experiment helps managers learn more about which factors are important and how they might affect the performance measures (responses). Metamodeling and optimization techniques can then be applied to seek a combination of factors that optimizes the decision objective. The cost data and allowed limit for each performance measure are listed in Table I.

6.2 Solution of the problem

The optimum design of the above maintenance float system is generated by using the proposed approach descibed in Sections 2 to 5:

- *Step 1 (define the maintenance float problem)*. The maintenance float system studied herein is illustrated in the previous subsection, the further details of this problem can be found in the literature (Madu and Kuei, 1996). Three performance measures including average equipment

Table I
Data for the illustrative maintenance float system

C_1=\$500/day	C_2=\$240/day	C_3=\$300/day	θ_1=0.9
θ_2=0.85	θ_3=0.5hr	θ_4=1.0hr	

utilization (*AEU*), average repair person utilization (*ASU*) and avergage waiting time for repair (*AWT*) are taken into consideration. The independent factors are number of independent and identical machine tools (*N*), number of standby machine tools (*F*), number of repair persons (*S*) and expected repair time (*R*). The dependent factors (interested simulation responses) include the three performance measures, *AEU*, *ASU* and *AWT*.

- *Step 2* (*construct the simulation model*). A simulation model is built for examination in the design of a maintenance float system using GPSS/H (Madu and Kuei, 1996).
- *Step 3* (*build the experimental design*). Taking the four controllable factors identified in Step 1, an orthogonal array is adopted to design the simulation experiment. The simulation runs will be conducted according to the various settings of controllable factors shown in Table II.
- *Step 4* (*run the simulation*). The simulation model is designed to yield information on the above three performance measures. The average realization of ten replications is used to represent a single run for each of the performance measures (i.e. 30 replications for each design point). The simulation results for each design point are summarized in Table II. Another set of simulation data shown in Table III is generated for validating the accuracy of the neural metamodel.
- *Step 5* (*construct the neural metamodel*). To construct the neural metamodel for the maintenance float system, a set of training data (refer to Table II) is obtained from the simulation experiments conducted in Steps 3 and 4. The four factors (*N*, *S*, *R*, *F*) are fed into the input nodes of neural network, and the three interested responses (*AEU*, *ASU*, *AWT*) are target values associated with output nodes. The BP-specific parameters are set as follows: learning rate = 0.1, momentum = 0.9,

Table II
Simulation results from experimental design for training data

Run	N	S	R	F	AEU	ASU	AWT
1	40	16	6	10	0.948	0.711	0.06
2	40	16	7	13	0.952	0.833	0.34
3	40	16	8	16	0.935	0.935	1.38
4	40	18	6	13	0.981	0.654	0.02
5	40	18	7	16	0.984	0.765	0.13
6	40	18	8	10	0.886	0.787	0.15
7	40	20	6	16	0.996	0.597	0
8	40	20	7	10	0.921	0.644	0.01
9	40	20	8	13	0.934	0.748	0.07
10	45	16	6	16	0.984	0.829	0.33
11	45	16	7	10	0.886	0.872	0.53
12	45	16	8	13	0.857	0.964	2.01
13	45	18	6	10	0.933	0.7	0.04
14	45	18	7	13	0.938	0.82	0.23
15	45	18	8	16	0.925	0.924	1.02
16	45	20	6	13	0.97	0.654	0.01
17	45	20	7	16	0.973	0.766	0.1
18	45	20	8	10	0.869	0.782	0.11
19	50	16	6	13	0.94	0.881	0.56
20	50	16	7	16	0.892	0.975	2.42
21	50	16	8	10	0.783	0.978	2.65
22	50	18	6	16	0.977	0.814	0.22
23	50	18	7	10	0.877	0.853	0.35
24	50	18	8	13	0.856	0.95	1.43
25	50	20	6	10	0.919	0.69	0.02
26	50	20	7	13	0.923	0.808	0.16
27	50	20	8	16	0.913	0.913	0.76

Table III
Simulation results for validation data

Run	*N*	*S*	*R*	*F*	*AEU*	*ASU*	*AWT*
1	47	17	6.0	14	0.969	0.803	0.20
2	42	16	7.4	11	0.899	0.873	0.55
3	48	18	8.0	13	0.873	0.931	1.06
4	43	20	7.4	12	0.926	0.737	0.06
5	42	17	7.2	15	0.960	0.854	0.42
6	41	19	6.0	11	0.960	0.621	0.01
7	45	16	6.2	14	0.964	0.842	0.37
8	43	18	6.4	10	0.927	0.708	0.05
9	49	19	7.6	15	0.919	0.900	0.65
10	44	17	8.0	16	0.915	0.947	1.15

learning epochs = 5,000, and the initial weights are randomly generated between [−0.3, 0.3].

Several neural network achitectures are run. The mean squared errors (MSE) of training data and validation data for various network architectures are summarized in Table IV. Observing from this table, the resulting information of function approximation by neural metamodel is more accurate than that of regression metamodel. The 4-5-3 architecture has comparably lowest MSE; the trained neural network of this architecture is taken as the simulation metamodel for the studied maintenance float system. Generally, the MSE of validation data is larger than that of training data because the network focuses on reducing the latter, not the former. From Figure 3, the validation error plummets and never goes up. It means the network with five hidden nodes does not have enough nodes to over-fit. The other network architectures in Table IV have similar behaviors.

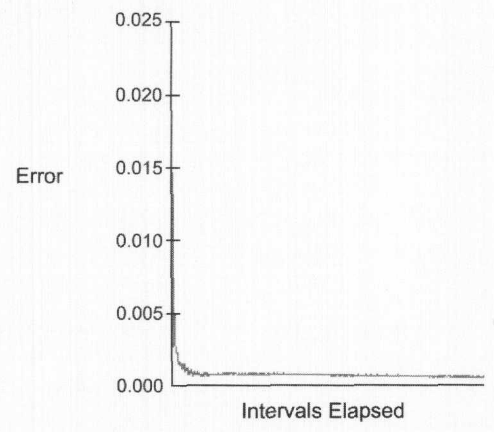
From Table IV, the 4-5-3 architecture is adopted to generate neural metamodel for this problem because it has the best predictive capability. The CPU time of learning for 4-5-3 architecture is 9.5s. In terms of MSE values of performance measures (*AEU*, *ASU* and *AWT*) shown in Table IV, the predictive power of neural metamodel is satisfactory. Comparing the results of regression and neural metamodels (refer to Table IV), the BP network outperforms regression in simulation metamodeling.

Since this study aims to optimize the maintenance float system design with minimal simulation runs, an orthogonal array $L_{27}(3^{13})$ is adopted to systematically and cautiously limit the training samples for neural networks. A leave-one-out method is suited to validate the accuracy of networks with small databases (Reich and Barai, 1999). The leave-one-out validation test is conducted by using the 27 patterns in Table II. The training and validation errors of these 27 leave-one-out validation tests are summarized in Table V. Since these 27 leave-one-out tests iteratively select 26 training patterns and one testing pattern, the training data does not unbiased cover the experimental space of the orthogonal design. Therefore, the validation errors of the 27 leave-one-out tests shown in Table V are comparably higher than the errors shown in Table IV. However, the result of validation error indicates that the neural network metamodel is credible. The results observed in these tables indeed can

Table IV
Prediction errors of training and validation for various network architectures

	Training MSE			Validation MSE		
	AEU	*ASU*	*AWT*	*AEU*	*ASU*	*AWT*
4-3-3	0.000102	0.000172	0.005575	0.000066	0.000028	0.005264
4-4-3	0.000037	0.000029	0.005009	0.000042	0.000077	0.001225
4-5-3	0.000015	0.000041	0.004306	0.000025	0.000019	0.001070
4-6-3	0.000016	0.000035	0.001478	0.000019	0.000022	0.001070
4-7-3	0.000048	0.000065	0.002377	0.000027	0.000193	0.007800
4-8-3	0.000013	0.000018	0.000389	0.000011	0.000062	0.007820
4-9-3	0.000005	0.000007	0.000177	0.000014	0.000034	0.008512
4-10-3	0.000005	0.000007	0.000135	0.000020	0.000038	0.007654
Regression	0.000228	0.000122	0.042439	0.000056	0.000179	0.025346

Figure 3
Error on validation data for the 4-5-3 architecture



demonstrate the effectiveness of neural metamodeling for the maintenance float system.

- *Step 6 (formulate the decision model)*. In Step 5, the neural metamodel is generated, and the nonlinear functions represent the relationships between the design variables and performance measures can thus be obtained. The decision model for the maintenance float system is formulated using the functions generated from the neural model and the data presented in Table I.
- *Step 7 (solve the decision model)*. The economic design of the maintenance float system can be obtained by solving the decision model, which stems from the neural metamodel. The proposed GA-based optimization method is adopted to solve the decision model. In this paper, the proposed GA has been implemented in C^{++}. The GA-specific parameters are set as follows: $Z = 100$ (population size), $T = 500$, $P_c = 0.95$, $P_m = 0.05$ and $b = 2$. After resolving the formulated decision model, the optimum combination of design variables involves 42 identical and independent machine tools, 16 standby machine tools, 14 repair persons, and the expected repair time is 6.93hr in the illustrative example. The minimum total cost (sum of the cost of lost production, the cost of repair persons and the cost of standby machines) of the maintenance float system is $9,134.80. The performance measures of average equipment utilization (*AEU*), average repair person utilization (*ASU*) and avergage waiting time for repair (*AWT*) are 0.95, 0.87 and 0.51hr, respectively. The proposed GA-based optimization algorithm requires 5.4s to generate this optimum design.

7. Concluding remarks

This paper demonstrates how simulation, neural metamodels and genetic algorithms can easily be combined to optimize the complicated maintenance float networks that are hard to be resolved analytically. The information of equipment availability and reliability will assist the planning and allocation of resources to perform essential tasks and services. The effective maintenance float system is constructed to minimize the cost of lost production incurred by the inherent unreliability in systems. This paper proposes an intelligent design approach to select the optimum combination of design parameters to achieve the desired performance measures in maintenance float systems at the minimum cost. The developed procedure incorporates simulation, neural networks and genetic algorithms. Neural networks are appropriate for constructing simulation metamodels. The resulting information of function approximation by neural metamodel is more accurate than that of regression metamodel. Using the GA-based optimization method, the proposed approach is also more practical than enumerating all possible simulation alternatives in the factors and objective function. From the illustrative example, the advantages of the developed procedure can be concluded as follows: first, the number of simulation runs required to generate an optimum design is reduced, and thus the computational requirement is significantly reduced; second, the procedure is simple and easy to implement; and third, it enables the

Table V
Summary of 27 leave-one-out validation tests of 4-5-3 network

	AEU	*ASU*	*AWT*
Training			
Average MSE	0.000017	0.000034	0.003676
Standard deviation MSE	0.000020	0.000043	0.008915
Validation			
Average MSE	0.000068	0.000073	0.010085
Standard deviation MSE	0.000182	0.000240	0.037708

complicated maintenance float problems to be resolved effectively.

References

Blanning, W.R. (1975), "The construction and implementation of metamodels", *Simulation*, Vol. 24-25 No. 6, pp. 177-84.

Bode, J. (2000), "Neural networks for cost estimation: simulations and pilot application", *International Journal of Production Research*, Vol. 38 No. 6, pp. 1231-54.

Chanin, M., Kuei, C.-H. and Lin, C. (1990), "Using Taguchi design, regression analysis and simulation to study maintenance float systems", *International Journal of Production Research*, Vol. 28 No. 1, pp. 11-19.

Chen, M.-C. (2001), "Tolerance synthesis using neural computing and nonlinear programming", *International Journal of Production Economics*, Vol. 70 No. 1, pp. 55-65.

Fogel, D.B. (1994), "An introduction to simulated evolutionary optimization", *IEEE Transactions on Neural Network*, Vol. 5 No. 1, pp. 3-14.

Gen, M. and Cheng, R. (1997), *Genetic Algorithms and Engineering Design*, John Wiley & Sons, New York, NY.

Goldberg, D.E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA.

Gross, D., Miller, D.R. and Soland, R.M. (1983), "A closed queueing network model for multi-echelon repairable item provisioning", *IIE Transactions*, Vol. 15 No. 4, pp. 344-52.

Holland, J.H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI.

Kleijnen, J.P.C. (1987), *Statistical Tools for Simulation Practitioners*, Marcel Dekker, New York, NY.

Kleijnen, J.P.C. and Sargent, R.G. (2000), "A methdology for fitting and validating metamodels in simulation", *European Journal of Operational Research*, Vol. 120 No. 1, pp. 14-29.

Kuei, C.H. and Madu, C.N. (1996), "Polynomial decomposition and Taguchi design for maintenance float system", *European Journal of Operational Research*, Vol. 72 No. 2, pp. 364-75.

Law, A.M. and Kelton, W.D. (1991), *Simulation Modeling and Analysis*, 2nd ed., McGraw-Hill, New York, NY.

Lin, C. (1996), "A practical approach to designing the maintenance system for a flexible manufacturing system", *International Journal in Computer Simulation*, Vol. 6 No. 3, pp. 379-91.

Lippmann, R.P. (1987), "An introduction to computing with neural nets", *IEEE ASSP Magazine*, Vol. 4, May, pp. 4-22.

Madu, C.N. (1988a), "Determination of maintenance floats using Buzen's algorithm", *International Journal of Production Research*, Vol. 26, pp. 1385-94.

Madu, C.N. (1988b), "A closed queueing network with two repair centers", *Journal of Operational Research Society*, Vol. 39 No. 10, pp. 959-67.

Madu, C.N. and Chanin, M.C. (1992), "A regression metamodel of a maintenance float problem with Erlang-2 failure distribution", *International Journal of Production Research*, Vol. 30 No. 4, pp. 871-85.

Madu, C.N. and Kuei, C.-H. (1996), "Multivariate simulation metamodels for a large-scale maintenance float system", *International Journal in Computer Simulation*, Vol. 6 No. 3, pp. 309-31.

Madu, C.N., Chain, M.N., Georgantzas, N.C. and Kuei, C.-H. (1990) "Coefficient of variation: a critical factor in maintenance float policy", *Computers and Operations Research*, Vol. 17 No. 2, pp. 177-85.

Mani, V. and Sarma, V.S. (1984), "Queueing network models for aircraft availability and spares management", *IEEE Transactions on Reliability*, Vol. R-33 No. 3, pp. 257-62.

Montgomery, D.C. (2000), *Design and Analysis of Experiments*, 5th ed., John Wiley & Sons, New York, NY.

Reich, Y. and Barai, S.V. (1999), "Evaluating machine learning models for engineering problems", *Artificial Intelligence in Engineering*, Vol. 13 No. 3, pp. 257-72.

Sahu, K.C. and Sharam, K.C. (1984), "Determination of optimal rotating float in a closed loop system: a case study", *International Journal of Production Research*, Vol. 22 No. 2, pp. 247-61.

Sarkar, M. and Yegnanarayana, B. (1998), "A review on merging some recent techniques with artificial neural networks", *Proceedings of the IEEE International Conference on Systems Man and Cybernetics*, pp. 1824-9.

Schaffer, J.D., Whitely, D. and Eshelman, L.J. (1992), "Combinations of genetic algorithms and neural networks: a survey of the state of the art", in Whitley, D. and Schaffer, J.D. (Eds), *Proceedings of the International Workshop on Combinations of Genetic Algorithms and Neural Networks, COGANN-92*, IEEE Computer Society Press, Washington, DC, pp. 1-37.

Smith, M. (1993), *Neural Networks for Statistical Modeling*, International Thomson Publishing, New York, NY.

Szczerbicki, E. (2000), "Simulation modeling for complex production systems", *Cybernetics and Systems*, Vol. 31 No. 3, pp. 333-51.

Zhang, H.-C. and Huang, H. (1995), "Applications of neural networks in manufacturing: a-state-of-the-art survey", *International Journal of Production Research*, Vol. 33 No. 3, pp. 705-28.

Further reading

Madu, C.N. and Georgantzas, N.C. (1988), "Waiting lines effects in analytical maintenance float policy", *Decision Sciences*, Vol. 19 No. 3, pp. 521-34.