Solving Bilevel Programming Problems Using a Neural Network Approach and Its Application to Power System Environment

Shamshul Bahar YAAKOB * and Junzo WATADA *

Abstract: In this paper, a hybrid neural network approach to solve mixed integer quadratic bilevel programming problems is proposed. Bilevel programming problems arise when one optimization problem, the upper problem, is constrained by another optimization, the lower problem. The mixed integer quadratic bilevel programming problem is transformed into a double-layered neural network. The combination of a genetic algorithm (GA) and a meta-controlled Boltzmann machine (BM) enables us to formulate a hybrid neural network approach to solving bilevel programming problems. The GA is used to generate the feasible partial solutions of the upper level and to provide the parameters for the lower level. The meta-controlled BM is employed to cope with the lower level problem. The lower level solution is transmitted to the upper level. This procedure enables us to obtain the whole upper level solution. The iterative processes can converge on the complete solution of this problem to generate an optimal one. The proposed method leads the mixed integer quadratic bilevel programming problem to a global optimal solution. Finally, a numerical example is used to illustrate the application of the method in a power system environment, which shows that the algorithm is feasible and advantageous.

Key Words: Boltzmann machine, meta-controlled Boltzmann machine, bilevel programming problem, mixed-integer quadratic problem.

1. Introduction

Bilevel programming has been successfully applied in diverse areas, including management science, economics, engineering, transportation [1],[2] and others since 1977, when Candler and Norton [3] first proposed its concept. Even today, bilevel programming has great application potential. However, due to its inherent non-convexity and non-differentiability, bilevel programming problems are often non-deterministic polynomial-time hard (NP-hard), and especially, nonlinear bilevel programming problem is very difficult to solve. A problem is NP-hard if it is polynomial-time reducible to an NP-complete problem. In the past few decades, efforts have been focused on developing effective algorithms for bilevel programming problems [4]–[6].

Certain limitations are recognized in one of the most eminent known algorithms to solve bilevel programming due to a local optimal solution, specific structures of bilevel programming, or Kuhn-Tucker conditions. The branch-and-bound method [7],[8], descent algorithms [9],[10], and an evolutionary method [11],[12] have been proposed for solving the bilevel programming problems. Additionally most of them require additional conditions for a solution. It is difficult to extend such a method to solve general nonlinear bilevel programming problems. Compared with classical optimization approaches, the prominent advantage of neural computing is that it can converge on the equilibrium point (optimal solution) rapidly, and this advantage has been encouraging researchers to solve bilevel programming problems using the neural network approach. Recently, Shih and Wen [13] and Lan and Wen [14] have proposed a neural network to solve the linear bilevel programming problem. Yaakob and Watada have proposed a double-layered hybrid neural network approach to solving mixed integer quadratic bilevel programming problem [15].

Furthermore, in real-world applications, the dynamic nature of the bilevel programming problems makes the problem very difficult to solve. Although various approaches have been suggested to reduce the burden of computation such as the use of fuzzy logic [16], dynamic programming [17] and interactive approaches [18], further research is still needed to obtain better approaches. A new type of neural network model called a meta-controlled BM is a promising technique to solve optimization problems because it can emulate the operations of the brain and uses parallel processing to save computational time.

A number of studies applied the bilevel programming approach to solve various problems in a power system environment. Carrion et al. [19] applied a bilevel programming approach to solve the medium-term decision-making problem faced by a power retailer. Arias et al. [20] presented a model for power systems operation planning formulated as a bilevel programming problem. Arroyo and Galiana [21] generalized a terrorist threat problem as a bilevel programming one.

The objective of this paper is to find the optimal combination investment in power systems. In this study, a hybrid meta-heuristic algorithm method is employed to resolve bilevel programming problems in a power system environment.

The remainder of the paper is organized as follows. Section 2 briefly explains the bilevel programming problem. Section 3 describes the proposed algorithm for solving bilevel programming problems. Results of a numerical example are reported in Section 4. Finally, Section 5 concludes the paper.
2. Bilevel Programming Problems

Bilevel programming is a case of multilevel mathematical programming that solves decentralized planning problems with multiple decision makers in a multilevel or hierarchical organization [12]. At different levels, decision makers play a game called the Stackelberg game [13], in which the follower responds to any decision made by a leader in the upper level of decision making but are not controlled directly by the leader. Thus, the leader is able to adjust the performance of the overall multilevel system indirectly by his decisions.

Bilevel programming involves two optimization problems where the constraint region of the first-level problem is implicitly determined by the other second-level optimization problem.

Let us describe mathematically the bilevel programming problem. A top level decision maker has control over the vector $x$, and a bottom level decision maker controls the vector $y$. Letting the performance functions of $F(x, y)$ and $f(x, y)$ for the two decision makers be linear and bounded, respectively, the bilevel programming problem can be written as

$$\max F(x, y) \quad \text{(Upper level)},$$

subject to $G(x, y) \leq 0$, where $y = y(x)$ is implicitly defined by

$$\min f(x, y) \quad \text{(Lower level)},$$

subject to $g(x, y) \leq 0$.

Obviously, the bilevel programming model consists of two sub-models, an upper-level and lower level model. $F(x, y)$ is the objective function of the upper level decision-makers, and $x$ is the decision vector of the upper level decision-makers; $G(x, y)$ is the constraint set of the upper level decision vector. $f(x, y)$ is the objective function of lower level decision-makers, $y$ is the decision vector of the lower level decision-makers, and $g(x, y)$ is the constraint set of the lower-level decision vector. The function $y = y(x)$ is usually called the reaction or response function.

The key idea for solving the bilevel programming model is to obtain the response function through solving the lower level problem and replace the variable $y$ in the upper level problem with the relationship between $x$ and $y$ – the response function. This response function connects the upper and lower level decision variables, which makes the two programming models dependent on each other. With only one optimizing variable, it could not optimize the whole system performance, which reflects the essence of the response function.

Compared with a conventional single-level programming model, the bilevel programming models have many more advantages. The main advantages are that (i) the bilevel programming can analyze two different and even conflicting objectives at the same time in the decision-making process, (ii) the multiple criteria decision-making methods of bilevel programming can more accurately predict real situations, and (iii) the bilevel programming method can explicitly represent mutual actions between two different levels of the decision makers.

Due to the investment problem in this study involving two kinds of decision-makers, those two levels are a strategic level decision making and an operational level decision making, which have a different objective function. Obviously, the bilevel programming model is appropriate to describe the investment problem. In this study, GA and a neural network (NN)-based method are employed to solve the bilevel programming problems.

3. Hybrid Metaheuristic Algorithm

The generic solution framework for the bilevel model in this study is an iterative three-phase process, in which the solution for either level depends on that for the other level. In the first phase, the partial solution of the upper level is generated, providing the parameters for the lower level. In the second phase, a meta-controlled BM is used to cope with the lower level on an NP-hard combinatorial optimization problem, giving the partial solution of the upper level. In the third phase, the solution of the lower level is transmitted to the upper level, the whole solution of the level is retrieved, and the upper-level objective is evaluated. The three phases proceed iteratively until the whole problem is maximally resolved.

3.1 Mathematical Formulation

A GA-based hybrid meta-heuristic algorithm was formulated, consisting of both Hopfield and Boltzmann machine neural networks and called meta-controlled BM. This hybrid metaheuristic algorithm allows us to solve bilevel programming problems. The hybrid metaheuristic model has two levels, which are referred to as the upper and lower levels. The functions of the levels are as follows:

1. The upper level will decide on the selection of the number of states and decide the portion of the selected states as a partial solution. Finally, after receiving feedback from the lower level, the upper level will decide the optimal/best solution.
2. The lower level will decide on the selection of the number of substations in selected states from the upper level and decide the portion of the selected substations.

The mathematical formulation for the bilevel model is written as follows:

**Upper level:**

$$\min T = R + r$$

where $R = \min \sum_{a=1}^{n} \sum_{b=1}^{n} \sigma_{ab} m_a \alpha_a m_b \beta_b$ (2)

subject to

$$\sum_{a=1}^{n} \mu_a m_a \alpha_a \geq P$$

$$\sum_{a=1}^{n} m_a \alpha_a = 1$$

$$\sum_{a=1}^{n} m_a = S$$

$$m_a \in \{0, 1\} \quad (a = 1, 2, ..., n)$$

$$\alpha_a \geq 0 \quad (a = 1, 2, ..., n)$$

Here, $T$ denotes a total risk from the upper level and lower level, $R$ is a risk calculated by GA at the upper level, and $r$ is the risk that is calculated at the lower level by meta-controlled BM. $P$ denotes the least acceptable rate of expected return, $\sigma_{ab}$ is the covariance between $a$ and $b$, $\mu_a$ is the expected return rate of $a$, $\alpha_a$ is the investment rate of $a$, $S$ denotes the desired number of units to be selected, and $m_a$ is 0-1 decision variable for $a$, where $m_a$ is 1 if any of $a$ is held and is 0 otherwise.

**Lower level:**

The lower level employs the meta-controlled BM where the objective function is converted into energy functions of two com-
ponents, namely, the Hopfield layer $E_h$, and the BM layer $E_{\text{lim}}$ as follows:

**Hopfield layer**

$$E_h = -\frac{1}{2} \sum_{i,j=1}^{n} \sigma_{ij} m_i x_i m_j x_j + \frac{1}{\beta} \sum_{i=1}^{n} \mu_i m_i x_i$$

(8)

**BM layer**

$$E_{\text{lim}} = -\frac{1}{2} \left( \sum_{i,j=1}^{n} \sigma_{ij} m_i x_i m_j x_j + 2 \sum_{i=1}^{n} \sum_{j=1}^{n} m_i x_i m_j x_j \right)$$

$$+ 2 \sum_{i=1}^{n} m_i x_i + \frac{1}{\beta} \sum_{i=1}^{n} \mu_i m_i x_i$$

(9)

Here, $\beta$ denotes a real number not less than 0, $\sigma_{ij}$ is the covariance between $i$ and $j$, $\mu_i$ is the expected return rate of $i$, $x_i$ is the investment rate of $i$, and $m_i$ is 0-1 decision variables for $i$, where $m_i$ is 1 if any of $i$ is held and $m_i$ is 0 otherwise. Here, $\beta$ is determined by $\alpha$ at the upper level and used as the weight of the expected return rates of the Hopfield and BM layers, respectively, and is the output of the $i^{th}$ unit of the Hopfield layer. The weight determines the relative importance of risk and expected return: a small value of $\beta$ means that maximizing expected return is more important than minimizing risk (“risk-taking” behavior), whereas a large value of $\beta$ means that risk should be minimized as much as possible, including the expense of decreasing return rate (“conservative” behavior).

### 3.2 Genetic Algorithm

The generation of the partial solution of the upper level must be considered carefully because it is the key to solving the whole problem efficiently. GA is a good alternative because of its robustness, flexibility, and relative simplicity. In this paper, GA is used to evaluate all the feasible solutions of the upper-level problem and thus ensures that the global optimum of the whole problem will be acquired if the lower level is addressed correctly.

The aim of the upper level is to select the limited number of states and decide the investment portion of the selected states that satisfy the decision makers. The mixed-integer portfolio selection problem is solved using a GA, as proposed by Watada et al. [22]. It is difficult to generate individuals who choose $N$ states out of $n$ states based on the binary encoding of a GA. Therefore, encoding is employed to set an allele to the number of binary variables and the string length to the number of choices. The selected states are decided based on a given the expected return rate and risk by a decision maker.

In general coding, the allele has $0, 1$ and the string length is $n$. However, the coding employed here makes the chromosome itself express the selected variable. In other words, in the coding that makes the allele have $0, 1$ and the string length $n$, each variable randomly is given the values “Selected” or “Not Selected” according to values “1” or “0”, respectively. Normally, there are $n$ non-zero values for $\alpha_e \geq 0$. If this exact solution is used, all feasible portfolios must receive investments, which is unrealistic because there are thousands of them. To obtain a more realistic solution, it is reasonable to fix, a priori, a small number $g$ and to invest in no more than $g$ different portfolios. If such a number is chosen, then, after a solution $\alpha_e$ is obtained, the reasonable way to choose these $g$ portfolios is to select the ones for which the recommended investing rates are the largest possible; then, the selected $g$ largest rates are normalized to guarantee that $\sum \alpha_e = 1$.

In this paper, because the number of selected variables is decided in advance, the selected numbers of variables are randomly coded. In this non-binary coding, the string length cannot exceed the selectable number because the string length is decided by the maximum selectable number of variables. Because the allele is decided by the total number of 0-1 variables, it does not produce an individual that selects no variable. Therefore, this non-binary coding is effective to produce an individual that is appropriate to the problem, in comparison with the binary coding in which the allele has $0, 1$ and the string length is $n$. If the total number of 0-1 variables is $n$ and the selectable number of the variables is $N$, then the following relation about the initial population and the new population obtained by a genetic operation is as follows:

$$n^N > 2^n \cdot N!$$

(10)

Then the non-binary coding using the allele with $\{1, 2, \ldots, n\}$ and the string length with $N$ is better able to produce better fitter individuals than the binary coding using the allele with $\{0, 1\}$ and the string length with $n$. Here, $n$ and $N$ are natural numbers and $n \geq N$.

The probability of generating individuals that select $N$ variables out of the total $n$ variables is $\frac{nC_N}{2n}$ when the allele is $\{0, 1\}$ and a string length is $N$. In this case, then the non-binary coding using the allele with $\{1, 2, \ldots, n\}$ and a string length of $N$ easily produces fitter individuals than the binary coding using the allele with $\{0, 1\}$ and the string length with $n$, if the following relation holds:

$$\frac{nC_N}{2^n} < \frac{nP_N}{n^N}$$

(11)

The above-mentioned relation can be written as follows:

$$\frac{nC_N}{2^n} < \frac{nP_N}{n^N}$$

$$\frac{n! \cdot n^N}{(n-N)! \cdot N!} < \frac{n! \cdot 2^n}{(n-N)!}$$

(12)

Therefore, equation (10) holds.

Figure 1 shows a relation between $n$ and $N$. If the selected number $N$ is approximately 1/3 of 0-1 variables, the non-binary coding with the allele with $\{1, 2, \ldots, n\}$ and a string length of $N$ easily produces an individual fitting the environment treated in the problem. If selected number $N$ is approximately 2/3 greater than number of 0-1 variables, $n$, the problem can be approached
by reversing the meaning in the problem. In other words, as in the case where the number of unselected 0-1 variables is less than about 1/3 of the total number of variables, this restriction can be solved by means of the compliment of the set of selected variables employed in the GA. This coding reduces stillbirths, which mean that an offspring is not fit for the environmental conditions given by a problem.

3.3 Meta-Controlled Boltzmann Machine

Conventionally, the number of units is determined on the basis of expert experience. To solve this problem, a double-layered neural network consisting of both Hopfield and Boltzmann neural networks is constructed. This double-layered model can be employed to select a limited number of units from those available. The double-layered model has two layers, which are referred to as the Hopfield and BM layers. The functions of the layers are as follows:

i. The Hopfield layer (Hopfield neural network) is used to select a limited number of units from the total and is called a "supervising layer".

ii. The Boltzmann layer (Boltzmann machine) is used to decide the optimal units from the limited number selected in the Hopfield layer and is called an "executing layer".

This meta-controlled BM is a new type of neural network model, which deletes units (neurons) in the BM layer that are not selected in the Hopfield layer during execution, as shown in Fig. 2. The BM layer of the meta-controlled BM is a function to be restructured using the selected units by the Hopfield layer. Because of this feature, the meta-controlled BM converges more efficiently than a conventional BM, which is an efficient method for solving a selection problem by transforming its objective function into the energy function because the Hopfield and Boltzmann networks converge at the minimum point of the energy function. The meta-controlled BM described above converts the objective function into energy functions of two components, namely, the Hopfield layer (Hopfield network) $E_h$ and the BM layer (Boltzmann machine) $E_{bm}$, as equations (8) and (9). The meta-controlled BM is tuned such that the Hopfield layer influences the BM layer with a probability of 0.9, and the BM layer influences the Hopfield layer with a probability of 0.1. Ultimately, experimentally, the meta-controlled BM is iterated with

$$U_i = 0.9y_i + 0.1x_i$$  \hspace{1cm} (14)

for the upper layer, and

$$L_i = x_i(0.9y_i + 0.1)$$  \hspace{1cm} (15)

for the lower layer. Here, $U_i$ in the upper layer is a value transferred to the other corresponding nodes of the upper layer, $L_i$ in the lower layer is a value transferred to other corresponding nodes of the lower layer, $y_i$ is a value of the present state at node $i$ of the upper layer; and $x_i$ is a value of the present state at node $i$ of the lower layer.

$L_i$ indicates that the value is influenced 90% by the value of node $i$ of the upper layer. When $U_i$ is 1, $L_i = x_i$. Otherwise, when $y_i$ is 0, the 10% value of $x_i$ is transferred to other nodes. Additionally, $U_i$ has 10% influences from the lower layer. Therefore, even if the upper layer converged on the local minimum, the disturbance from the lower layer makes the upper escape the local minimum. When the local minima have large barriers, dynamic behavior may change by changing 0.9 and 0.1 dynamically. This phenomenon is similar to simulated annealing. The main reason for selecting these two probabilities is that, based on trial and error, it is found that the selected probabilities provide the best solution to the proposed method.

3.4 Hybrid Metaheuristic Algorithm

The proposed algorithm is summarized as follows:

Step 1. Start with a series of randomly generated combinations. Then GA will select a set potential solution combination. Pass the combination to the lower level.

Step 2. Set each parameter to its initial value for meta-controlled BM.

Step 3. Input $1/\beta$ as a weight for the Hopfield layer and BM layer.

Step 4. Execute the Hopfield layer.

Step 5. If the output value of a unit in the Hopfield layer is 1, add some amount of this value to the corresponding unit in the BM layer. Execute the BM layer.

Step 6. After executing the BM layer at a constant frequency, decrease the temperature.

Step 7. If the output value is sufficiently large, add a certain amount of the value to the corresponding unit in the Hopfield layer.

Step 8. Iterate from Step 4 to Step 7 until the temperature reaches the restructuring temperature.

Step 9. Restructure the BM layer using selected units in the Hopfield layer.

Step 10. Execute the BM layer until it reaches the termination condition. Transmit the calculated solution to the upper-level.

Step 11. The whole solution of the upper level is retrieved, and the upper level objective is evaluated.

Step 12. The processes proceed interactively until the whole problem is maximally resolved. List all the individual solutions.
4. Numerical Examples

In this section, a numerical example is presented to show the effectiveness of the proposed method. In this simulation, a decision maker first decides, based on the states indices, how much should be invested into different states and then decides, within each state, which substation to select. The parameters of the corresponding method are given in Table 1.

<table>
<thead>
<tr>
<th>Table 1 Conditions of the simulation.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper level</strong></td>
</tr>
<tr>
<td>Offspring/reproduction rate</td>
</tr>
<tr>
<td>Crossover rate</td>
</tr>
<tr>
<td>Mutation rate</td>
</tr>
<tr>
<td>Population/Group size</td>
</tr>
<tr>
<td><strong>Lower level</strong></td>
</tr>
<tr>
<td>Initial value temperature</td>
</tr>
<tr>
<td>End value temperature</td>
</tr>
<tr>
<td>Operation frequency</td>
</tr>
<tr>
<td>Number of units</td>
</tr>
</tbody>
</table>

In the empirical study, as shown in Table 2, investment portfolios over fifteen states were analyzed and optimized based on ten-year failure rates. Table 3 shows the annual failure number for each substation in each state. SS.D1 denotes a substation D1. The proposed effective investment strategy considers both, the expected failure rate, measured by the mean of the given failure data, and the risk, measured by the variance. The analysis of the trade-offs between mean and variance of the failure numbers employs mean-variance analysis and is implemented using the proposed method. Table 4 gives the partial solution from the upper level, providing the parameters for the lower level. The partial solutions shown in Table 4 are the results calculated for all combinations of the states. In this paper, based on the minimal risk, only the five best combinations of investment were listed. Focusing on the upper level in the simulations, in the case of a fixed crossover rate and mutation rate, the increase of the selection rate has a tendency to decrease the average time of search. Therefore, the average search rate decreases with increasing selection rate. Thus, the increase improves its efficient search solution. Nevertheless, the lower the number of the population is, the earlier the generations in which the optimum solution can be obtained is. Because a large population makes the search space wide and large, the average time of search and the average rate of search become worse, even though the time of reaching the optimum solution is almost ten generations. Therefore, the trade-off relation holds between increasing the selection rate and decreasing the population size.

In the case where only a mutation rate is changed with a fixed crossover rate and selection rate, the increase of the mutation rate has a tendency to decrease the average time of search. Therefore, the average search rate decreases with increasing mutation rate. Thus, the increase improves its efficient search solution. Nevertheless, the lower the number of the population is, the earlier the generations in which the optimum solution can be obtained is. Because a large population makes the search space wide and large, the average time of search and the average rate of search become worse, even though the time of reaching the optimum solution is almost ten generations. Therefore, the trade-off relation holds between increasing the selection rate and decreasing the population size.

In the case where only a mutation rate is changed with a fixed selection rate and crossover rate, the increasing mutation rate increases the average rate of search. Therefore, increasing the mutation rate results in an increase in the average rate of search. As a result, reaching the optimal solution tends to occur more often according to the increase of the mutation rate as well as the increase of the crossover rate.

According to simulation results of a hundred generations on each combination of parameters, the average generation when the optimal individual appears is found relatively early. Especially when the population size is large, the optimum solution is found within ten generations.

In the reality, the problem is often where the total selected unit has been decided in advance or provided within a certain range. Thus, the combination of coding method and genetic operators proposed in this study to solve the upper level is capable of obtaining the solution effectively when the GA method is applied to the real problem.

Table 5 provided the results of the lower level. SS.A1 denotes a substation A1. It shows the selected substations in each state, based on the upper level partial solutions. After deciding how much money to invest in each state, the substations to invest from that state’s substations must be selected. As an example, consider the case of Comb.1 and apply this method to selecting a state’s substations. Four substations should be selected over the ten substations available. A meta-controlled BM method was applied to select four substations from ten substations with the highest investing rates, as given in Table 5.
example, in State A, SS.A1, SS.A3, SS.A4 and SS.A5 have been selected. In this paper, the results for only the five best combinations are shown. The same calculations will be performed on other combinations. Table 6 presents the improvement of the solution given by the proposed method. R denotes the risk obtained from upper level, and r denotes the risk obtained from lower level. T represents the total of the risk. It shows that the final ranking list of combination Comb.1 is the optimal solution with a risk value of 0.004278 followed by Comb.3 with a risk of 0.004559. Comb.2 is 0.004908. Comb.5 is 0.004903 and Comb.4 is 0.004908.

In this study, it is clear that the proposed method improves the optimum solution in comparison with the conventional method as shown in Table 7. The conventional method presented herein used the enumeration method for the upper layer and BM to solve the lower layer problem. The proposed method can give a better and near-optimal solution both in the upper layer and in the lower layer. The results also showed that the objective values improved significantly compared to the conventional method. Although it is generally accepted that the conventional approaches are suitable to solve this kind of problems, the results in this paper suggest that the proposed method is outperforms the conventional method. Even in this paper, more emphasis is placed on the quality solution, but the computational time must also be considered. We have planned to apply grid computing/parallel computing, so that the computational time will be reduced.

The simulation results show that the proposed method can be used to analyze two different objectives simultaneously during the decision making process. The results from lower layer give the combination of the state, and the upper layer gives the combination of the substations in the selected state. Thus, it is a decision process with multi value criteria, which is more realistic. The interaction between strategic level and the operational level can be interpreted clearly.

Table 6 Optimal solution.

<table>
<thead>
<tr>
<th>Comb.</th>
<th>R</th>
<th>r</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.001061</td>
<td>0.003217</td>
<td>0.004178</td>
</tr>
<tr>
<td>2</td>
<td>0.001258</td>
<td>0.003301</td>
<td>0.004559</td>
</tr>
<tr>
<td>3</td>
<td>0.001235</td>
<td>0.003555</td>
<td>0.004790</td>
</tr>
<tr>
<td>4</td>
<td>0.001731</td>
<td>0.003172</td>
<td>0.004903</td>
</tr>
<tr>
<td>5</td>
<td>0.001487</td>
<td>0.003421</td>
<td>0.004908</td>
</tr>
</tbody>
</table>

Table 7 Comparison with the conventional method.

<table>
<thead>
<tr>
<th></th>
<th>Conventional method</th>
<th>Proposed Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>0.001072</td>
<td>0.001061</td>
</tr>
<tr>
<td>r</td>
<td>0.003229</td>
<td>0.003217</td>
</tr>
<tr>
<td>T</td>
<td>0.004301</td>
<td>0.004278</td>
</tr>
</tbody>
</table>


5. Conclusions

A new formulation of investment allocation based on bilevel programming was proposed in this paper. The proposed algorithm was used to solve examples of bilevel programming problems for two-level management investment in a power system environment. The numerical examples showed that the hybrid neural network approach is effective and practical.

A case study of the investment allocation in several states was presented. The bilevel formulation was used to distinguish the two different problems, one of the strategic levels and the other of the operational level. The proposed method provides a feasible alternative solution. In the future, the proposed method will be further investigated to enhance its quality.

Acknowledgments

The first author would like to thank Universiti Malaysia Perlis and the Ministry of Higher Education Malaysia for a study leave at Waseda University under the SLAI-KPT scholarship. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the quality of this paper.

References


Shamshul Bahar Yaakob

He received his Bachelor’s degree in Computer Engineering in 1991 from Shizuoka University, Japan and M.Eng. degree in Electrical and Electronic Systems in 1996 from the Nagaoka University of Technology, Japan. He is a Lecturer in the School of Electrical System Engineering, Universiti Malaysia Perlis, Perlis, Malaysia, and is currently pursuing his Ph.D. degree at Waseda University. His research interests include soft computing, reliability optimization, and multi-criteria optimization. He is a student member of IEEE.

Junzo Watada (Member)

He received his B.S. and M.S. degrees in Electrical Engineering from Osaka City University, Japan, and his Ph.D. degree from Osaka Prefecture University, Japan. Currently he is a Professor of Management Engineering, Knowledge Engineering and Soft Computing at the Graduate School of Information, Production & Systems, Waseda University, Japan. He was the recipient of the Henri Coanda Medal Award from Inventico in Romania in 2002, and is a Fellow of SOFT. His professional interests include metaheuristics, tracking systems, knowledge engineering, and management engineering.