Use of Neural Networks in Bridge Management Systems

HOSNY A. MOHAMED, A. O. ABD EL HALIM, AND A. G. RAZAQPUR

One of the most urgent problems related to highway infrastructure is that the cost of maintaining a network of bridges with an acceptable level of service is more than the available budgeted funds. Low prioritization of the available resources allocated to bridge projects exacerbates the situation. About 42 percent of the 574,000 highway bridges in the United States were reported by FHWA to be structurally deficient or functionally obsolete. Traditional management practices have become inadequate as ways to face this serious problem. Priority-setting schemes for bridge projects range from those done on a subjective basis in which engineering judgment is used to those that use very complex optimization models. However, currently used priority-setting schemes do not have the ability to optimize the system's benefits to obtain optimal solutions. The present objective is to show how artificial neural networks (ANNs) can be used to optimize the system's resources to generate the group of bridge improvements that minimizes the loss of the network benefits. ANNs are algorithms with characteristics that are able to solve certain classes of optimization problems. The advantages of using ANNs include improvements in the speed of operation by parallel implementation either in hardware or in software. It is also possible to implement ANNs by optical devices that operate at higher speeds than traditional electronic chips.

Bridges and pavements represent the major investment in a highway network. In addition, they are in constant need of maintenance, rehabilitation, and replacement. The main problem facing most transportation agencies is that the cost of maintaining the bridge network with an acceptable level of service is more than the available budgeted funds. About one-half of the 574,000 highway bridges in the United States were built before 1940, and 42 percent of them were reported by FHWA to be structurally deficient or functionally obsolete (*1*). This finding forced many states to start developing bridge management systems (BMSs).

BMSs are a relatively new approach developed after the successful application of the systems concept to pavement management. A BMS would organize and carry out the bridge projects to meet the needs of the network. The primary objective of a BMS is to integrate all bridge activities into a comprehensive computerized system such that the most efficient and cost-effective performance is achieved (*2*).

At present the cost of rehabilitation and replacement of bridges consumes most of the funding available for bridge improvements. Setting priorities to carry out these activities represents the most challenging task of the BMS. Priority-setting schemes for bridge projects range from those done on a subjective basis in which engineering judgment is used to those that use very complex optimization models. However, the available schemes can be grouped into four types: sufficiency rating, level-of-service (LOS) deficiency ranking, incremental benefit–cost analysis, and mathematical programming. The first three types calculate a ranking index for each project and then sort all projects in descending order of their indexes. Starting with the project with the highest ranking index, projects will be carried out until the available funds are exhausted. Those techniques could provide good solutions, but not the optimal ones. Mathematical programming techniques can provide better decisions and have been used in BMSs by Pontis and North Carolina. However, it is not the intent of this paper to critique priority-setting schemes for bridge projects. Mohamed (*3*) has provided evaluations and comparisons of the available schemes. The objective of this paper is to show how artificial neural networks (ANNs) can be used to optimize the system's resources to generate the group of bridge improvements that minimize the loss of the network benefits.

Department of Civil and Environmental Engineering, Carleton University, Ottawa, Ontario, Canada K1S 5B6.

ARTIFICIAL NEURAL NETWORKS

An ANN takes after its biological analog through its composition of nodes and the connections among them. The first attempt to simulate neural networks was made in 1943 by McCulloch and Pitts (*4*). The basic idea of ANNs is to construct a network of cells that are called artificial neurons, nodes, units, or processing elements (PEs). The synapses of biological networks are simulated by weighted connection. Figure 1 shows the structure of a single PE in a network. The ith PE receives input (X_j) from the jth PE. The arrows in Figure 1 represent the input connection from other PEs. The weight of each connection (T_{ij}) is analogous to the strength of the synaptic connection between neurons. The PE has only a single output that can be input into many other PEs. The net input into the ith PE can be written as follows (*5*):

$$\text{Input}_i = \sum_j X_j \cdot T_{ij} \quad (1)$$

After estimating the net input, it will be converted to an activation value, $\text{Act}_i(t)$, which is

$$\text{Act}_i(t) = F_i\,[\text{Act}_i(t-1), \text{Input}_i(t)] \quad (2)$$

Equation 2 describes the activation value as a function of the net input, and it may also depend on the previous value of activation, $\text{Act}_i(t-1)$. By applying the output function to the activation value, the output (X_i) of the artificial neuron (i) can be obtained as follows:

$$X_i = f_i\,(\text{Act}_i) \quad (3)$$

Several types of neural networks exist. Each type has a different architecture, activation function, output function, and weighted connections. These characteristics depend on the function of the

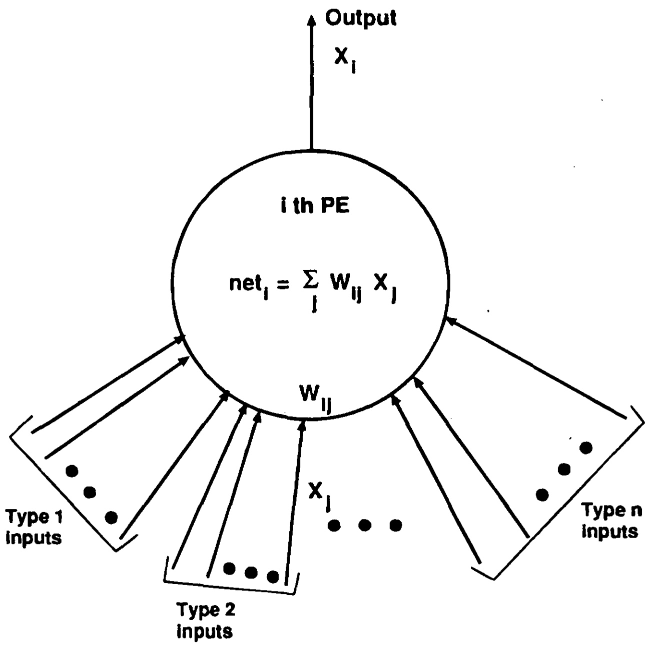

FIGURE 1 Structure of single processing element (5).

human brain that the ANN aims to simulate. Such functions include but are not limited to the following (*6*): prediction and estimation, pattern recognition, clustering, and optimization.

Although ANNs have proven to be useful tools in a variety of problem-solving areas, there is relatively little research or practical applications of them in the field of transportation engineering. As a part of a comprehensive research plan to develop a BMS, an ANN was developed to allocate a budget to bridge projects. This paper describes the network that was developed and its application to bridge management.

MODEL FORMULATION

Because the bridge problem has two dimensions (the time dimension and the network dimension) a dynamic programming model has been developed to handle the time dimension. The bridge network is simulated by an ANN. Formulation of the dynamic programming model is beyond the scope of this paper, but Mohamed (*3*) provides details of this model. The objective of the model is to allocate the available budget to bridge projects to minimize the loss of the systems benefits. Budget allocation to bridge projects includes choosing the best improvement alternative for each bridge in the network and determining the optimum timing for carrying it out. The general objective function of the model was formulated as follows:

$$\text{Minimize } Z = \sum_{t=1}^{T} \sum_{B=1}^{N} \sum_{A=0}^{m(t,B)} \text{BL}(t,B,A) \cdot X(t,B,A) \quad (4)$$

where

z = the total loss of system benefits,
T = analysis period (in years),
N = total number of bridges,
$m(t,B)$ = number of improvement alternatives for bridge B in year t,
$\text{BL}(t,B,A)$ = the amount of loss of system benefits [benefit loss (BL)] if alternative A for bridge B was chosen in year t and
$X(t,B,A)$ = 1 if alternative A for bridge B was chosen in year t and 0 otherwise.

The BL for each alternative can be estimated by the summation of two parts; the first part is the agency BL, and the second part is the increased user cost due to LOS deficiencies (*1*). The input of the model requires information about all possible alternatives for all bridges with the associated life-cycle costs (LCCs) and increased user costs (IUCs) if the alternative were to be implemented in any year of the analysis period. Engineering expertise is needed to determine the possible alternatives to be considered for each bridge in the network. Estimation of life cycle costs and increased user costs for all of the proposed alternatives is required. The BL associated with carrying out any improvement alternative can be estimated from the following formula:

$$\text{BL}(t,B,A) = \{[\text{LCC}(t,B,A) - \text{LCC}(t,B,E) + \text{IUC}(t,B,A)]\} \cdot (P/F, r, t-1) \quad (5)$$

where

$\text{LCC}(t,B,A)$ = life-cycle cost of alternative A for bridge B if carried out in year t;
$\text{IUC}(t,B,A)$ = increased user costs due to LOS deficiencies associated with alternative t,B,A;
E = the alternative for bridge B in year t that has minimum (LCC + IUC); and
$(P/F, r, t-1)$ = present worth factor for real rate of return r and $(t-1)$ years.

It should be noted that for the do-nothing alternative, BL will be calculated as the present worth of delaying the E alternative for 1 year plus the extra costs of doing nothing, such as posting the bridge. This is because if a bridge was not chosen in year t it will be considered in the optimization of the following year (i.e., (t + 1).

After omitting the time, which is the responsibility of the dynamic programming model, the objective function of the bridge network problem and its constraints can be written as follows:

$$\text{Minimize } Z = \sum_{B=1}^{N} \sum_{A=0}^{m(B)} \text{BL}(B,A) \cdot X(B,A) \quad (6)$$

subject to

$$\sum_{B=1}^{N} \sum_{A=0}^{m(B)} \text{IC}(B,A) \cdot X(B,A) \leq W \quad (7)$$

$$\sum_{A=0}^{m(B)} X(B,A) = 1 \text{ for all B} \quad (8)$$

$$X(B,A) = 1, 0 \quad (9)$$

where

Z = the benefit loss for any year t,
N = total number of bridges,

$m(B)$ = number of improvement alternatives for bridge B,
$IC(B,A)$ = initial cost if alternative A was chosen for bridge B,
$BL(B,A)$ = the loss in system benefits if alternative A was chosen for bridge B,
$X(B,A)$ = 1 if alternative A was chosen for bridge B and 0 otherwise, and
W = the available budget (dollars/year).

To solve the optimization problem associated with the developed dynamic model the neural network technique has been adopted, as discussed in the following section.

DEVELOPED ANN

From the operations researcher's point of view, ANNs are algorithms with certain characteristics that can be used to solve certain optimization tasks. The advantages of using ANNs include improvements in the speed of operation by parallel implementation either in hardware or in software. Therefore, neural networks can do well even on conventional computers. It is also possible to implement ANNs by optical devices, which operate at higher speeds than traditional electronic chips.

The common approach to the construction of optimization neural networks is to formulate the problem in terms of minimizing a cost or energy function. This approach is known as the Hopfield network (*7*). A modified Hopfield network will be used to construct the proposed ANN. Two main steps should be followed in mapping an optimization problem onto a neural network that uses an energy function (*8*): (a) choose a network architecture that decodes neurons' outputs into a solution to the problem, and, (b) formulate an energy function that generates the best solutions at its minima.

Energy functions resemble penalty functions in operation research. The objective function and the problem constraints will be included in the energy function as follows (*9*):

$$E = \sum_i v_i \,(\text{violation of constraint } i) + u\,(\text{cost}) \quad (10)$$

where

E = energy function,
v_i, u = energy function parameters (always > 0), and
cost = an optimization cost function that depends on the problem.

NETWORK STRUCTURE

The ANN developed for the present study is basically a Hopfield network, but with a dynamic penalty parameter. The approach used to map this kind of network was presented by Wang and Chankong (*10*). This network was developed after testing another kind of network with constant penalty parameters. The latter has failed to converge to a stable state, however. Figure 2 shows the network architecture. The network consists of two layers. The first one has two neurons, corresponding to L and ΔL, which will be defined later. The second layer has $n+1$ massively connected neurons, where $n = N \cdot m$, where N is the total number of bridges and m is the number of improvement alternatives for each bridge. It should be noted that the do-nothing alternative is included in the m alternatives.

Each neuron will receive four inputs, and these are feedback input from itself, input from all other neurons, input from neurons representing alternatives for the same bridge, and the benefit loss due to the alternative that the neuron represents. The costs of improvement alternatives will represent the weights of the neuron connections.

The energy function is represented by the following equation:

$$E = Z - L\,[g(x)] \quad (11)$$

where

$$Z = \sum_{i=1}^{n} BL_i \cdot X_i \quad (12)$$

$$g(x) = \frac{1}{2}\left\{\sum_{i=1}^{n+1} IC_i \cdot X_i) - W)^2 + \frac{1}{2}\left[\sum_{j=1}^{N}\left(\sum_{i=m(j-1)+1}^{i=mj} X_i\right) - 1\right]^2\right\} \quad (13)$$

where L is the penalty parameter and $X(B,A)$ is equal to X_i. It should be noted that Z is the original objective function introduced in Equation 6. The first term of the penalty function $g(x)$ represents the budget constraint, whereas the second term represents the constraints of Equation 8, which ensure that for every bridge one and only one alternative should be selected. The form of Equation 13 will ensure that the penalty function $g(x)$ will have its lowest value when all of the constraints are satisfied; otherwise, it will be greater. A slack variable, $X(n+1)$, has been introduced to convert the budget inequality constraint to an equality constraint.

The penalty parameter L will be decreasing by an amount ΔL, which is inversely proportional to the violation of constraints:

$$\Delta L(t) = 1/g[x(t)] \quad (14)$$

$$L(t+1) = L(t) - [\Delta t \cdot \Delta L(t)] \quad (15)$$

where Δt is equal to step size (0.001 to 1.0) and $L(0)$ is the initial penalty parameter (0.0001 to 1.0).

The net input of each neuron will be changed by an amount equal to the change of the energy function due to changing the state of this neuron divided by ΔL. The division by ΔL will change the net input by an amount that is proportional to the violation of constraints. The net input for each neuron will be estimated by the following equation:

$$\text{Input}_i\,(t+1) = \text{Input}_i\,(t) - \Delta t \cdot \left[\frac{\partial E}{\partial X_i} \Big/ \frac{\Delta L(t)}{a}\right] \quad (16)$$

where

$$\frac{\partial E}{\partial X_i} = BL_i - L\left\{IC_i \cdot \left[\left(\sum_{j=1}^{n} IC_j \cdot X_j\right) + X_{n+1} - W\right] + \sum_{j=1}^{n} X_j - N\right\} \text{ for } i = 1 \text{ to } n \quad (17)$$

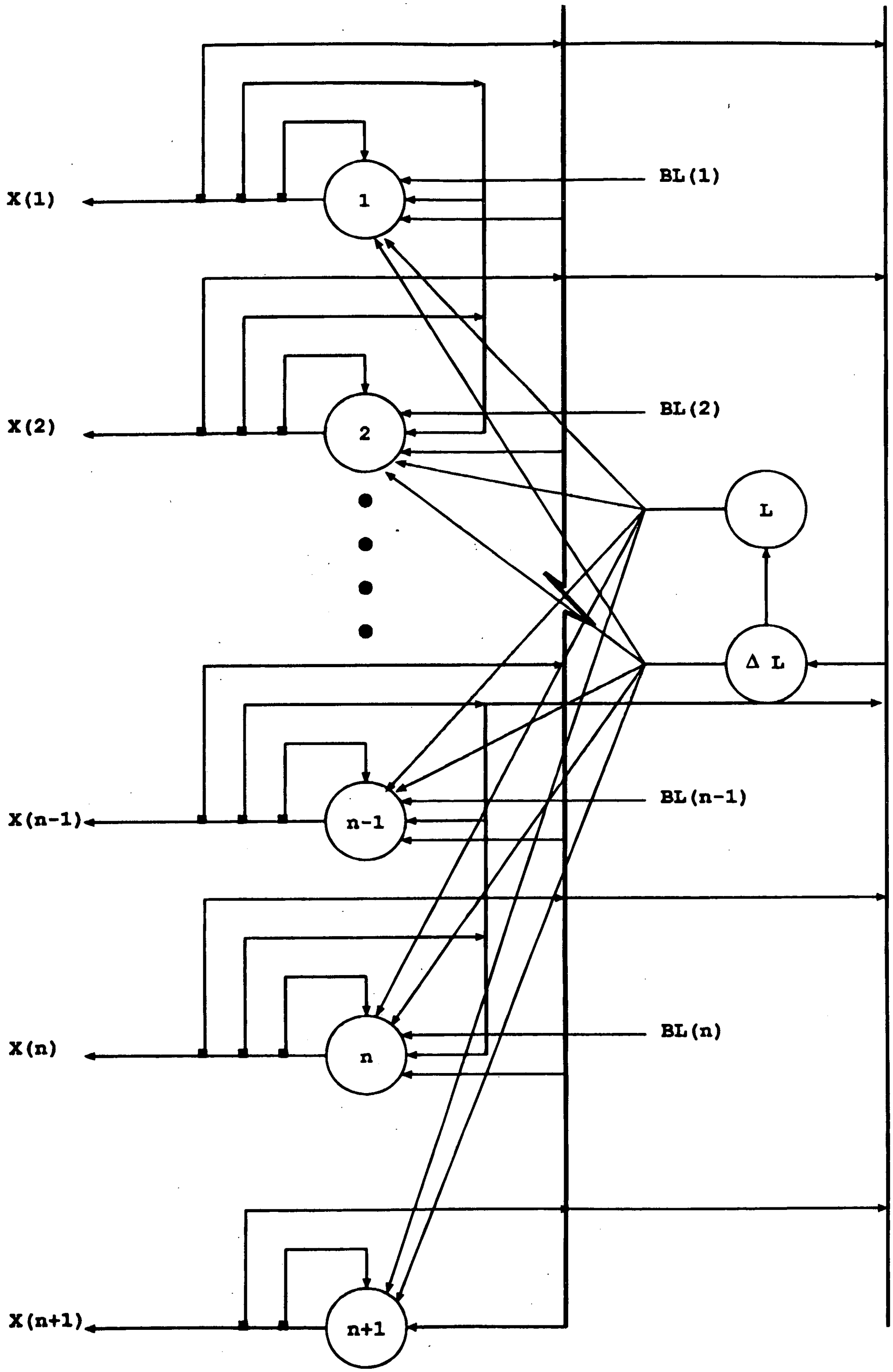

FIGURE 2 Neural network structure.

TABLE 1 Data for Example 1

Bridge	Alternative	Benefit Loss	Initial Costs	Solution
1	1	200,000	600,000	0
	2	500,000	300,000	1
2	1	300,000	500,000	1
	2	700,000	200,000	0

$$\frac{\partial E}{\partial X_{n+1}} = -L\left[\left(\sum_{i=1}^{n+1} \mathrm{IC}_i \cdot X_i\right) - W\right] \tag{18}$$

and where a is an adjusting parameter (10 to 100) and input (0) is the initial state (assumed to be zero).

The activation function $\mathrm{ACT}_i\,(t)$ will be taken as the following deterministic sigmoid function (5):

$$\mathrm{Act}_i\,(t) = \frac{1}{1 + e^{-\mathrm{Input}_i\,(t)/q}} \tag{19}$$

where q is a positive scaling constant.

By applying the output function $X_i\,(t)$ to the activated value, the state or the output of the neuron can be updated as follows:

$$X_i(t) = 0 \qquad \text{if } \mathrm{Act}_i(t) \leq \epsilon \tag{20}$$

$$X_i(t) = 1 \qquad \text{if } \mathrm{Act}_i(t) \geq (1 - \epsilon) \tag{21}$$

$$X_i\,(t) = \mathrm{Act}_i\,(t) \text{ otherwise} \tag{22}$$

where ϵ is the permissible error, which can be set between 0.01 and 0.1.

The initial states of all neurons will be set to zero. The network will be supplied by the BL and the IC of each alternative for all bridges and the available budget (W). Each neuron will begin to send impulses to other neurons through synapses or connections until the network reaches a stable state. A stable state means that each neuron is either on or off; in other words, the output X_i of each neuron is either 1 or zero. When the value of X_i is 1, this alternative will be carried out. On the other hand the value of zero corresponds to canceling this alternative. If a bridge will not receive any improvement the do-nothing alternative will be on.

APPLICATION OF DEVELOPED NETWORK

In this section the application of the developed neural network is demonstrated by using artificial data. A network simulator was constructed by using Turbo Pascal. The code for this program is available on request. Physical implementation of the proposed neural network in a parallel distribution fashion would result in significant computation enhancement. The performance of the simulated network will be demonstrated through two examples.

Example 1

In Example 1 the network was fed data about two bridges, and each bridge had two improvement alternatives. The available budget was \$800,000, and the BLs and ICs were as given in Table 1. Figures 3 and 4 illustrate the convergent patterns of the objective function (Z)

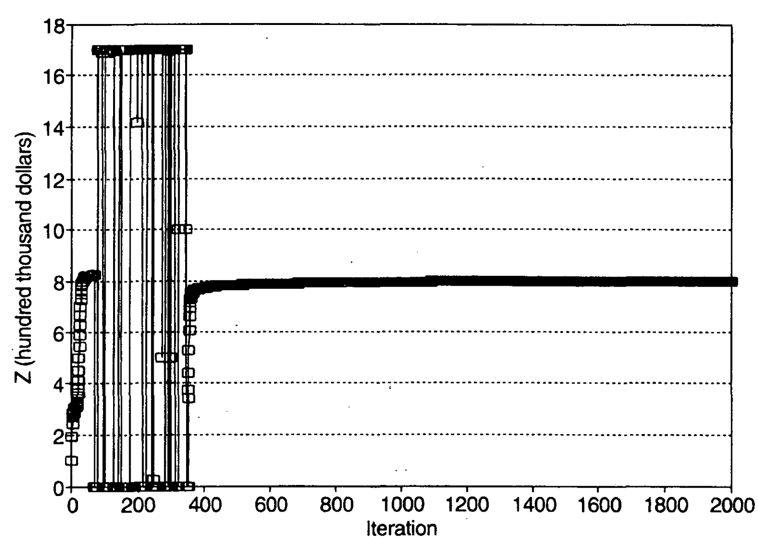

FIGURE 3 Convergent patterns of objective function Z in Example 1.

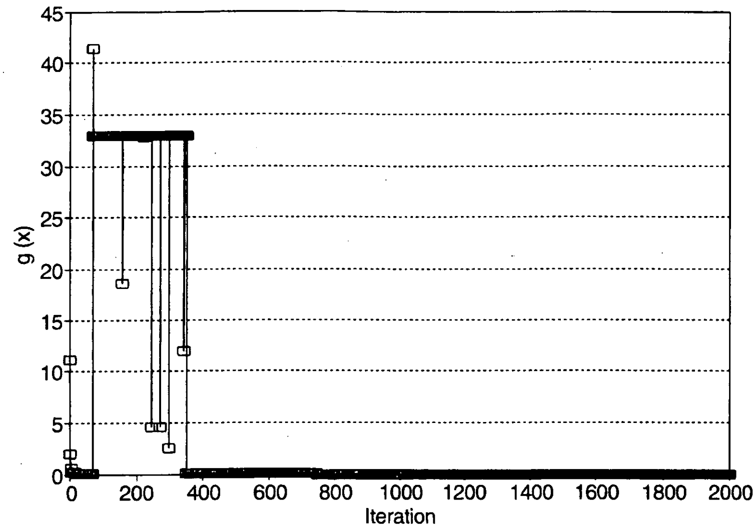

FIGURE 4 Convergent patterns of penalty function $g(x)$ in Example 1.

and the penalty function [$g(x)$], respectively. The optimal solution for this problem is to carry out alternatives 1-2 and 2-1 with a minimum Z of $800,000. It can be seen from Figure 3 that after 100 iterations the network reached the optimum, but it later diverged for about 300 iterations before reaching its stable state. From Figure 4 it can be noticed that between the 100 and the 400 iterations the constraints were violated, because $g(x)$ values were greater than zero. After 400 iterations the network converged and all neurons reached stable states; one for 1-2 and 2-1 and zero for the others. The network selected the best alternatives, which generated the minimum benefit losses, satisfied the budget constraints, and had only one neuron on for each bridge.

Example 2

Example 2 illustrates the ability of the network to find the optimal decision for a network of 10 bridges. Each bridge has two improvement alternatives, and the budget was limited to $3,200,000. A commercial software package for integer programming called DSS was used to find the optimal solution for this problem before running the network simulator. The latter program found the solution in 135 sec, whereas the developed ANN took only 20 sec. The network was able to converge after 6,000 iterations. The network data and the solution are given in Table 2. The objective function at the optima will have a value of 44 hundred thousand. Figure 5 shows

TABLE 2 Data for Example 2

Bridge	Alternative	Benefit Loss	Initial Costs	Solution
1	1	200,000	600,000	0
	2	500,000	300,000	1
2	1	300,000	500,000	1
	2	700,000	200,000	0
3	1	400,000	200,000	1
	2	700,000	200,000	0
4	1	200,000	600,000	0
	2	500,000	300,000	1
5	1	200,000	600,000	0
	2	500,000	300,000	1
6	1	200,000	600,000	0
	2	500,000	300,000	1
7	1	300,000	500,000	1
	2	700,000	200,000	0
8	1	400,000	200,000	1
	2	700,000	200,000	0
9	1	200,000	600,000	0
	2	500,000	300,000	1
10	1	200,000	600,000	0
	2	500,000	300,000	1

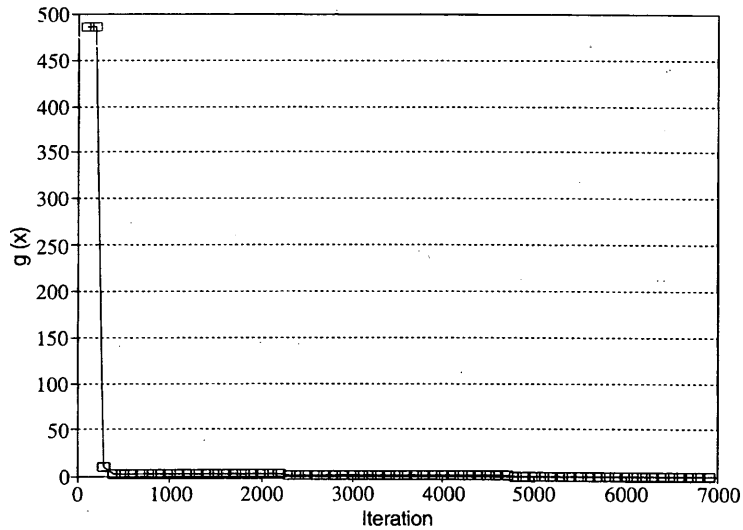

FIGURE 5 Convergent patterns of penalty function $g(x)$ in Example 2.

that the constraints were satisfied after about 500 iterations, and then a search was conducted for the optima until the network converged, as shown in Figure 6.

SUMMARY AND CONCLUSIONS

The low prioritization of budget dollars allocated to bridge activities has led to an imbalance between the needs of a bridge network and fiscal constraints. The cost of rehabilitation and replacement of bridges consumes most of the funding available for bridge improvements. Setting priorities to carry out these activities represents the core of the BMS. Existing priority-setting schemes for bridge projects can provide good solutions but not the optimal decisions for allocating funds to bridge activities. A neural network was developed to allocate a budget to bridge projects in a specific year. The architecture of the developed network reduces the memory storage space required for the computer and improves the speed of operation by parallel implementation either in hardware or in software.

More specifically, the analysis and results presented in this paper lead to the following conclusions:

1. There is an urgent need for an efficient scheme to set priorities for bridge management.
2. The bridge problem has two dimensions. The time dimension can be modeled by dynamic programming, whereas the network dimension can be simulated by a neural network.
3. ANNs can be used to allocate a budget to bridge projects.
4. The ANN that was developed has the potential to be used to allocate funds for large numbers of bridges with unlimited viable alternatives.

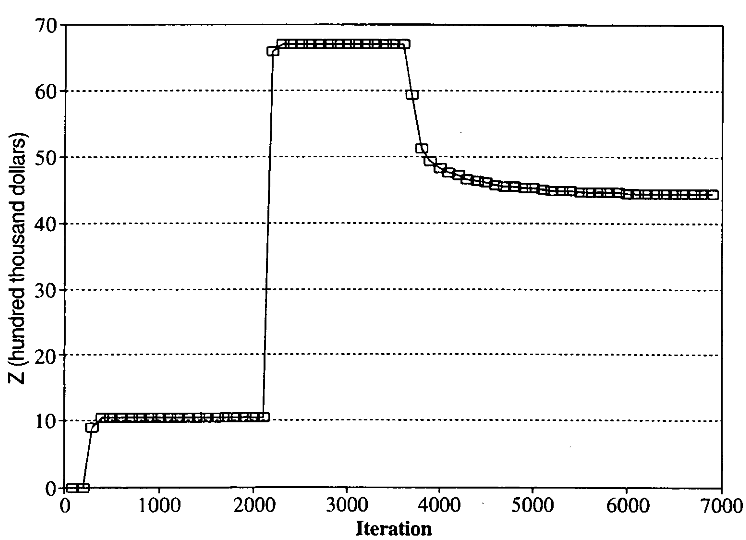

FIGURE 6 Convergent patterns of objective function Z in Example 2.

The network that was developed needs to be tested on real data for a bridge network. The feasibility and performance of the neural network for large bridge networks will be examined when data for such networks are available.

ACKNOWLEDGMENTS

The authors acknowledge the financial assistance provided by the Natural Sciences and Engineering Research Council of Canada (NSERC Scholarships) and the Transportation Association of Canada (TAC Scholarships). The authors also thank the following people for technical advice: George Hibbert of Public Works Canada and R. Hafez of the Department of Systems and Computer Engineering at Carleton University.

REFERENCES

1. *Bridge Management Systems.* Report FHWA DP-71-01R. FHWA, U.S. Department of Transportation, 1989.
2. Mohamed, H. A., A. O. Abd El Halim, and A. G. Razaqpur. Bridge Management Systems: Conceptual Framework. *Proc., Al-Azhar Engineering Third International Conference,* Cairo, Egypt, Dec. 1993.
3. Mohamed, H. A. *Development of Optimal Strategies for Bridge Management systems.* Ph.D. thesis proposal. Department of Civil and Environmental Engineering, Carleton University, Ottawa, Ontario, Canada, 1994.
4. McCulloch, W. S., and W. Pitts. A Logical Calculus of the Ideas Imminent in Nervous Activity. *Bulletin of Mathematical Biophysics,* Vol. 5, 1943, pp. 115–133.
5. Freeman, J. A., and D. M. Skapura. *Neural Networks: Algorithms, Applications, and Programming Techniques.* Addison-Wesley Publishing Company, Inc., Reading, Mass., 1991.
6. Burke, L. I., and J. P. Ignizio. Neural Networks and Operation Research: An Overview. *Computers & Operation Research,* Vol. 19, No. 3/4, 1992.
7. Hopfield, J. J., and D. W. Tank. Neural Computation of Decisions in Optimization Problems. *Biology and Cybernetics,* Vol. 52, 1985, pp. 141–152.
8. Ramanujam, J., and P. Sadayappan. Optimization by Neural Networks. In *Proc., IEEE Second International Conference on Neural Networks,* Vol. II, 1988, pp. 325–332.
9. Looi, C.-K. Neural Network Methods in Combinatorial Optimization. *Computers & Operation Research,* Vol. 19, No. 3/4, 1992.
10. Wang, J., and V. Chankong. Recurrent Neural Network for Linear Programming: Analysis and Design Principles. *Computers & Operation Research,* Vol. 19, No. 3/4, 1992.

Publication of this paper sponsored by Committee on Structures Maintenance and Management.