Development of a Neural Network for Optimization and Its Application to Assembly Line Balancing

*Dae Sun Hong , **Byoung Jae Ahn, , *Joong Ho Shin and *Won Jee Chung

*Dept. of Mechanical Design and Manufacturing Engineering, Changwon National University, Korea
(Tel:+82-55-279-7577, Fax:+82-55-263-5221, e-mail:dshong@sarim.changwon.ac.kr)
*Denso PS Corporation, Changwon, Korea
(Tel:+82-55-279-9500, e-mail:fogman@empal.com)

Abstract: This study develops a neural network for solving optimization problems. Hopfield network has been used for such problems, but it frequently gives abnormal solutions or non-optimal solutions. Moreover, it takes much time for solving a solution. To overcome such disadvantages, this study adopts a neural network whose output nodes change with a small value at every evolution, and the proposed neural network is applied to solve ALB (Assembly Line Balancing) problems . Given a precedence diagram and a required number of workstations, an ALB problem is solved while achieving even distribution of workload among workstations. Here, the workload variance is used as the index of workload deviation, and is reflected to an energy function. The simulation results show that the proposed neural network yields good results for solving ALB problems with high success rate and fast execution time.

Keywords: neural network, optimization, assembly line balancing

1. INTRODUCTION

Assembly line balancing (ALB) problems aim to allocate assembly tasks to a number of workstations so that idle time should be minimized while satisfying the precedence and cycle time constraints. The problems are great concern to researchers , and much research effort has been made[1]. The methods for solving these problems are categorized into two approaches: analytical approach[2] and heuristic approach[3]. The analytical approach can find exact solutions, but they become intractable with large size problems[1]. On the other hand, the analytical approach can cope with large size ones, but they do not guarantee an optimal solution.

Meanwhile, the Hopfield neural network[4,5] has been applied to solve various optimization problems including ALB problems[6,7]. However, this method has a problem of frequently generating a non-optimal or meaningless solution because of converging to a local minimum.

To overcome such disadvantage, this study proposes a neural network for optimization, then it is applied to solve ALB (assembly line balancing) problems. In this study, the objective is to achieve even distribution of workload between workstations while minimizing the idle time. Modifying the Hopfield neural network, the proposed neural network is designed such that each output is changed with a small value at every evolution according to the sign of their change.

To show the effectiveness, the proposed neural network is applied to solve ALB problems. The application results show that the proposed neural network is fast, and has a high rate of yielding optimal solutions.

2. NEURAL NETWORK FOR OPTIMIZATION

2.1 Line Balancing Problem

An assembly line is assumed to consist of a number of serially connected workstations in which N assembly tasks are adequately allocated to a number of workstations. Each assembly task is given a process time t_j. Then, the work content T_{wc}, total assembly time for completing all tasks, is defined by

$$T_{wc} = \sum_{1}^{N} t_j \quad . \tag{1}$$

An ALB solution can be represented by a set of M workstations each of which ws_i is given a workload t_i, the sum of process times for the assigned tasks. The assembly tasks need to be allocated to M workstations such that workload deviation among workstations should be minimized.

As a measure of workload deviation, the workload variance V and the mean absolute deviation of workloads MAD are introduced[8], and are defined by

$$V = \frac{1}{M}\sum_{i=1}^{M}\left(t_i - \bar{t}\right)^2 \tag{2}$$

$$MAD = \frac{1}{M}\sum_{i=1}^{M}\left| t_i - \bar{t} \right| \tag{3}$$

where $\bar{t}$ is the mean workstation time evaluated by

$$\bar{t} = \frac{T_{wc}}{M}. \tag{4}$$

In this study, the workload variance V is reflected to an energy function of the neural network for deriving an evolution equation.

2.2 The Proposed Neural Network

This study develops a neural network based on modification of the Hopfield-Tank network for optimization

[5]. The evolution equation of the network is expressed by

$$net_i = \sum_{j=1}^{n} w_{ij} v_j - u_i + I_i \tag{5}$$

$$\frac{du_i}{dt} = net_i \tag{6}$$

$$v_i = g(u_i) \tag{7}$$

where v_i (i=1,2, .,,n. n is the number of output nodes) is the output of a neuron ($0 \le v_i \le 1$) , u_i is the total input to the i-th neuron, w_{ij} is the interconnection weight between two nodes, I_i is the external input, and $g(u_i)$ is a sigmoid function.

Eqs (5)-(7) are highly nonlinear differential equations, and it takes much time to numeric ally compute them. It can be observed from simulation that an output node tends to change in a large amount once in a while. In this case, the network usually converges to a local minimum to generate a non-optimal solution.

To overcome such problems, this study proposes a neural network in which the change of output is restricted to a small value according to the sign of Eq. (5). That is,

$$u_i(k+1) = u_i(k) + \Delta u_i \tag{8}$$

where

$$\Delta u_i = \begin{cases} -c & \text{if } net_i < 0 \\ 0 & \text{if } net_i = 0 \\ c & \text{if } net_i > 0 \end{cases} \tag{9}$$

where k is a time step and c is a small positive constant. Fig. 1 shows a block diagram for the evolution of the proposed neural network, based upon Eqs (5) and (7)- (9).

The energy function of the network can be expressed by [4]

$$E = E_{net} + \sum_{i=1}^{n} \int_0^{v_i} g(x)^{-1} dx \tag{10}$$

where E_{net} is the network energy expressed by

$$E_{net} = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} v_i v_j - \sum_{i=1}^{n} I_i v_i \,. \tag{11}$$

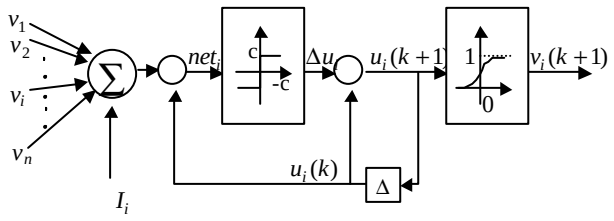

Fig. 1 Block diagram for the evolution of the proposed neural network

Based upon differentiating the above equation with respect to v_i, the following relationship is derived [7]:

$$net_i = -u_i - \frac{\partial E_{net}}{\partial v_i} \,. \tag{12}$$

The above equation is to be extended for solving ALB problems adopting an NxM network.

2.3 Neural Network for ALB Problems

To apply the proposed neural network for solving ALB problems, and Eqs (10) and (12) are modified as follows, respectively:

$$E = E_{ALB} + \sum_{i=1}^{N} \sum_{a=1}^{M} \int_0^{v_{ia}} g(x)^{-1} dx \tag{13}$$

$$net_{ia} = -u_{ia} - \frac{\partial E_{ALB}}{\partial v_{ia}} \tag{14}$$

where E_{ALB} is an energy function for assembly line balancing, i=1,2, .., N, and a =1,2, …, M. (N is the number of tasks and M is the number of workstations).

This study defines the E_{ALB} to consist of five energy terms as in the followings:

$$E_{ALB} = E_1 + E_2 + E_3 + E_4 + E_5 \,. \tag{15}$$

E_1 and E_3 are terms for abnormal solution, E_2 the penalty term for violation of the precedence constraint, E_4 is the one for requirement of workcontent, and E_5 is the one for the deviation of workload

E_1 is defined in such a manner that it increases when more than a task allocated to more than one workstation, and can be expressed by

$$E_1 = \frac{A}{2} \sum_{i=1}^{N} \sum_{a=1}^{M} \sum_{\substack{b=1 \\ b \ne a}}^{M} v_{ia} v_{ib} \tag{16}$$

where A is a positive constant.

E_2 is the energy term for violation of precedence constraints, and is expressed by [6]

$$E_2 = \frac{B}{2} \sum_{a=1}^{M} \sum_{b=1}^{M} \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} q_{ab} v_{ia} v_{jb} \tag{17}$$

where B is a positive constant. q_{ab} is an index according to the order of workstations as in the followings:

$$q_{ab} = \begin{cases} 1: \text{ for } a > b \\ 0: \text{ otherwise} \end{cases} \tag{18}$$

a_{ij} is an index for the precedence constraint b etween two tasks, and is defined by

$$a_{ij} = \begin{cases} 1: \text{if tasks } i \text{ and } j \text{ satisfy} \\ \quad \text{the precedence constraint} \\ 0: \text{otherwise.} \end{cases} \quad (19)$$

E_3 is the penalty term that increases when the number of tasks expressed in the network is not equal to N, and is expressed by

$$E_3 = \frac{C}{2}\left(\sum_{i=1}^{N}\sum_{a=1}^{M} v_{ia} - N\right)^2 \quad (20)$$

where C is a positive constant.

E_4 is the penalty term which increases when the total workcontent expressed in the network is not equal to the one in Eq.(1), and is expressed by

$$E_4 = \frac{D}{2}\left(\sum_{i=1}^{N}\sum_{a=1}^{M} t_i v_{ia} - T_{wc}\right)^2. \quad (21)$$

where D is a positive constant.

E_5 is the term for workload deviation for all workstations, and is defined by

$$E_5 = \frac{F}{2}\sum_{a=1}^{M}\left(\bar{t} - \sum_{i=1}^{N} t_i v_{ia}\right)^2. \quad (22)$$

where F is a positive constant.

From summing Eqs (16) to (22), E_{ALB} is obtained by

$$\begin{aligned} E_{ALB} = & \frac{A}{2}\sum_{i=1}^{N}\sum_{a=1}^{M}\sum_{\substack{b=1 \\ b\neq a}}^{M} v_{ia} v_{ib} \\ & + \frac{B}{2}\sum_{a=1}^{M}\sum_{b=1}^{M}\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} q_{ab} v_{ia} v_{jb} \\ & + \frac{C}{2}\left(\sum_{i=1}^{N}\sum_{a=1}^{M} v_{ia} - N\right)^2 \\ & + \frac{D}{2}\left(\sum_{i=1}^{N}\sum_{a=1}^{M} t_i v_{ia} - T_{wc}\right)^2 \\ & + \frac{F}{2}\sum_{a=1}^{M}\left(\bar{t} - \sum_{i=1}^{N} t_i v_{ia}\right)^2 \end{aligned} \quad (23)$$

Finally, based on Eq. (14), the evolution equation is derived by

$$\begin{aligned} net_{ia} = & -u_{ia} - A\sum_{j\neq i}^{M} v_{ja} - B\sum_{y=1}^{N}\sum_{j=1}^{M} a_{ij} q_{iab} v_{jb} \\ & - C\left(\sum_{j=1}^{N}\sum_{b=1}^{M} v_{jb} - N\right) \\ & - Dt_i\left(\sum_{i=1}^{N}\sum_{b=1}^{M} t_i v_{jb} - T_{wc}\right) \\ & - Ft_i\sum_{b=1}^{M}\left(\bar{t} - \sum_{j=1}^{N} t_j v_{jb}\right). \end{aligned} \quad (24)$$

The right term of the above equation is then calculated once per each revolution for verifying the sign of net_{ia}, then the change of output is evaluated as in the followings:

$$\Delta u_{ia} = \begin{cases} -c & \text{if } net_{ia} < 0 \\ 0 & \text{if } net_{ia} = 0 \\ c & \text{if } net_{ia} > 0 \end{cases} \quad (25)$$

$$u_{ia}(k+1) = u_{ia}(k) + \Delta u_{ia} \quad (26)$$

$$v_{ia}(k) = g(u_{ia}(k)) \quad (27)$$

The above equation is to be used for solving ALB problems.

3. CASE STUDY

To show the effectiveness of the proposed approach, it is applied to solve ALB problems, and are compared with Hopfield network approach.

3.1. Application to ALB Problem

In this study, ALB problems are given a precedence diagram and the number of workstations. Fig. 2 shows an example of precedence diagram an eight-assembly-task problem. In the figure, the number in a node represents task number, and the number below each node does process time for the task. This example shows four-workstation balancing solution in which the assembly tasks inside a dotted line constitute a workstation.

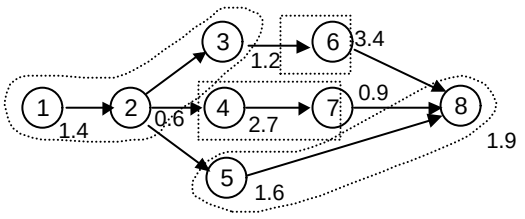

Fig. 2. Eight-assembly-task problem and the optimal solution for four workstations.

The program is written in Visual C++ executed by a PC with 300MHz CPU. In this simulation, the constants in Eq. (24) are chosen as A=B=20 and C=D=F=5, while constant c in Eq. (25) is 2.

The above problem is represented by an NxM network: N=8, M=4. Fig. 2 shows an example of evolution of the network. In the figures, each row represents a part number, and each column is a workstation.

In the figure, the size of rectangle is depicted to be proportional to the value of output between zero and unity. The initial values are randomly assigned with small values, and the network evolves by Eq. (24) to yield a solution. Fig. 3(c) shows that workstation 1 consists of tasks 1, 2 and 3, workstation 2 consists of tasks 2 and 7, and etc. In this case, the cycle time is shown to be 3.6 sec.

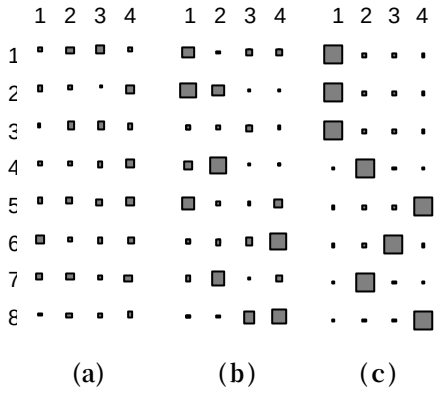

Fig. 3. Evolution of the network during a simulation run.
(a) Initial pattern (b) k=400, (c) k=3000

To show the performance, the proposed method is applied to solving Jackson's 11-task problem[9] shown in Fig. 5, and the result is shown in Table 1 to be compared with the one of Ref. [8].

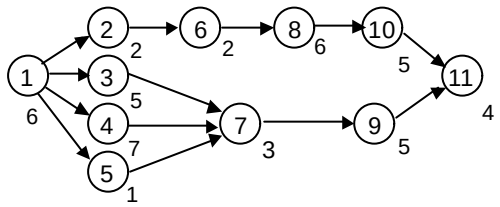

Fig. 4. Precedence diagram for Jackson's 11-assembly-task problem.

Table 1 shows the simulation results in which the numbers of workstations are given 8, 6, 5, 4, and 3s. For each number of workstations, 30 simulation runs were carried out, and yield the cycle time and *MAD*. The *MAD*'s obtained are compared with those in Ref. [8]

The table shows that the proposed approach successfully solves ALB problems, and the workload deviations are smaller than those of Ref. [8]; the MAD are reduced by 35.4% in average.

Table 1. Results for Jackson's 11-task problem.

No. of workstations	Cycle time	MAD (proposed)	MAD[8]
8	7	0.8125	1.0000
6	9	0.6228	1.0000
5	11	0.7200	-
4	12	0.5000	-
3	16	0.4444	0.8889

3.2. Comparison with Hopfield-Tank network

To compare the performance, the proposed approach and Hopfield network [5] are used for solv ing the ALB problem of Fig.2, respectively. The optimal solution of the problem is known. In this simulation, 30 simulation runs were carried out for each method. Table 2 shows the comparison result.

Table 2. Performance comparison.

Performance	Proposed approach	Hopfield network
Rate of finding a feasible solution (%)	76.0	33.3
Rate of finding an optimal solution (%)	43.3	10.0
Execution time (sec)	10	1500

It can be seen from the table that the proposed approach has a superior performance to the Hopfield network. The rate of finding a feasible solution is increased by more than two times, and that of finding an optimal solution is increased by more than four times. Especially, the execution time for the proposed network is only one fifteenth of the Hopfield network.

From the case study, it is concluded that the proposed neural network approach can successively solve ALB problems, and the performance of the network is much better than Hopfield network.

4. CONCLUSION

This study proposed a neural network for optimization, and it is applied to solve ALB problems. The results are summarized as follows:

(1) Modifying the Hopfield neural network, this paper proposes a neural neural network in which each output is changed with a small value at every evolution according to the sign of the net input.
(2) The workload variance among workstations is reflected to an energy function of the neural network for deriving an evolution equation
(3) The proposed neural network is applied to solve ALB problems, and the result shows that it yields the solution having small workload deviation.
(4) In comparison of the Hopfield network, the proposed approach has a superior performance in execution speed and solution quality to the Hopfield network.

From the case study, it is concluded that the proposed neural network for optimization can successively solve ALB problems, and shows a high performance.

ACKNOWLEDGEMENTS

This work was supported by Changwon National University and the Korea Science and Engineering Foundation (KOSEF) through the Machine Tool Research Center at Changwon National University.

REFERENCES

[1] S. Ghosh and R. J. Gagran, "A Comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems", *Int. J. Prod. Res*. 27, pp. 637-670. 1989.

[2]. I. Baybars "A survey of exact algorithms for simple assembly line balancing problem", *Management Sci* Vol.32, No.8, pp.909-932. 1986.

[3] F.B. Talbot and J.H. Gehrlein, "A comparative evaluation of heuristic line balancing techniques", *Management Science*, Vol.32. No.4, pp.430-454, 1986.

[4] J. J. Hopfield. "Neural networks and physical systems with emergent collective computational abilities", *Proc. of National Academy of Scientist*, Vol. 79, pp. 558, 1982.

[5] J. J. Hopfield and D. W. Tank. "neural computation of decisions in optimization problems", *Biol. Cybern.*, Vol.52, pp. 141-152, 1985.

[6] Y. Hashimoto, I. Nishikawa, T. Watanabe, and H. Tokumaru, "Line balancing problems using a Hopfield network", 1369-1375, *Japan-USA Symp on Flexible Automation*, Japan, 1994.

[7] D.S. Hong and H.S. Cho, "A neural-network-based computational scheme for generating optimized robotic assembly sequences", *Engineering Applications of Artificial Intelligence*, Vol. 8, No. 2, pp.129-145, 1995.

[8]. R. Rachamadugu and B. Talbot, "Improving the quality of workload assignment in assembly lines", *Int. J. Prod. Res*., vol.29, No.3, pp.619-633, 1991.

[9] J.R. Jackson, "A computing procedure for a line balancing problem", *Management Science*, Vol.2, No.3, pp.261-271, 1956