Artificial Intelligence for Engineering Design, Analysis and Manufacturing (1998), *12*, 273–287. Printed in the USA.
Copyright © 1998 Cambridge University Press 0890-0604/98 $12.50

An artificial neural network for resource leveling problems

NABIL KARTAM[1] AND TANIT TONGTHONG[2]

[1]Department of Civil Engineering, Kuwait University, P.O. Box 5969, Safat, 13060 Kuwait.
[2]Department of Civil Engineering, Vongchavalitkul University, Muang, Nakhonratchasima, Thailand.

(RECEIVED April 2, 1997; ACCEPTED December 16, 1997)

Abstract

A new methodology for solving resource leveling problems is introduced using Artificial Neural Networks (ANN). This paper describes a new efficient and robust approach which is unique to those utilized by traditional heuristic and optimization resource leveling techniques. The Resource Leveling Artificial Neural Network (RLANN) exploits advantages of both Hopfield networks and competition-based artificial neural networks. The universal scheme of the RLANN is applicable to construction project networks produced with Critical Path Method (CPM), in forms of either arrow or precedence diagrams. The scheme is comprised of two layers, an input and a competition layer, of artificial node matrices fully connected by links. Solving mechanisms inside the RLANN are based on an equation of motion and a competition strategy that control the level of daily resource usage. While the equation of motion governs activities to be shifted within schedule constraints, the competition process finds the best positions for the activities to achieve optimum results. The approach is simple and can be implemented on either a personal computer or a parallel processing device. The solutions produced are comparable to, or better than, those generated by other heuristic or optimization techniques. This paper describes the development of the RLANN, its solving mechanisms, and its uses in construction resource leveling problems. The comparison of the result of the RLANN to those of other traditional techniques is also included. The conclusions highlight the applicability of this model to other civil engineering problems.

Keywords: Artificial Neural Network; Resource Allocation; Resource Management

1. INTRODUCTION

In general, the completion date and cost of a construction project are major constraints: By contract, the project must be completed by a specific date and within a certain budget. Project managers need to find a schedule and management techniques that establish an effective plan to execute the activities and utilize the resources efficiently. Without a schedule reflecting proper resource management, delay of project completion and cost overruns can occur.

A project schedule produced with CPM alone is not adequate because the availability of resources is not considered while generating the schedule. Demands for resources on a specific time may be found unnecessarily high and/or undesirable fluctuations in resource utilization may occur. Resource scheduling and resource leveling are two basic techniques for improving project schedules for a more realistic presentation of resource utilization (Antill & Woodhead, 1982; Moder et al., 1983). Resource scheduling is applied when limits on the quantities of particular resources are fixed. The technique shifts project activities such that the demands for the resources at a period of time are under the resource limits while minimizing the extension of project duration. Resource leveling, on the other hand, is applied when unlimited resources are available. The leveling technique reorganizes noncritical activities to reduce fluctuations in daily resource utilization without altering the activities' logical sequences nor prolonging project duration.

Resource leveling is a primary technique which provides project managers a determination of the minimum amount of resources necessary to accomplish a project by a specified completion date. The obtained minimum demand then indicates whether it exceeds the limit of resource availability, if additional resources are required and in what quantity, or if the resource scheduling technique is needed. The other two objectives of the leveling technique are to mini-

Reprint requests to: Nabil Kartam, Associate Professor, Department of Civil Engineering, Kuwait University, P.O. Box 5969, Safat, 13060 Kuwait.; E-mail: kartam@kuc01.kuniv.edu.kw

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

mize the peak of daily resource requirements and to reduce overall project costs by decreasing day-to-day fluctuation in resource usage and lessening resource idleness (Harris, 1978). Smoothness of resource utilization rate implies good management in efficient assigning continuous work to resources. Low productivity due to releasing and hiring resources is decreased as well.

A heuristic approach (Burgess & Killebrew, 1962; Galbreath, 1965; Shaffer et al., 1965; Harris, 1978, 1990; Antill & Woodhead, 1982; Ahuja, 1984; Martinez, 1993) and an optimization approach (Phillips & Garcia-Diaz, 1981; Antill & Woodhead, 1982; Moder et al., 1983; Easa, 1989; Shah et al., 1993) are two traditional resource leveling techniques. Without searching through all possible paths, the heuristic approach uses some predetermined rules and procedures to reschedule activities in order to obtain a satisfactory solution. However, no heuristic method is always better than the others because a particular heuristic can provide a superior solution in one specific project but a worse result in another. As a result, to come up with a near-optimum schedule for a construction project remains uncertain. The optimization approach, on the other hand, employs some mathematical techniques such as Linear Programming (LP) and enumeration method to find optimum schedules. This approach performs well in a small-sized problem. In large-scale networks which are common for construction projects, combinatorial explosion of alternatives makes this approach impractical due to the unpredictable computation times and limited computer capacity.

The research addressed in this paper primarily proposes a new, simple, and effective approach which can be a functional alternative to solve resource leveling problems. The Resource Leveling Artificial Neural Network (RLANN) is developed and applied to level resources in various project schedules. The RLANN augments the capabilities of Competition-Based Artificial Neural Networks (CBANNs) with those advantages of Hopfield networks: the minimization of an energy function and the derivation of connection weights. The organization of this paper will be as follows. First, the paper describes various competition-based ANNs. Next, it presents the development of the RLANN including its structure, problem formulation, energy function, and equation of motion. This paper then presents the solving mechanism and competition strategies inside the RLANN. The use of the model and the experiment results are discussed. The conclusions identify the merits and further extension of the proposed model.

2. COMPETITION-BASED ANNS VERSUS HOPFIELD NETWORKS

There are many varieties of CBANN models including Kohonen's self-organizing feature map (Kohonen, 1988), Elastic net (Durbin & Willshaw, 1987), Angeniol's net (Angeniol et al., 1988), and Grossberg's competitive learning neural network (Grossberg, 1987). Among these models, some have shown success in several optimization problems (Fang & Li, 1990). In general, the basic CBANN consists of an input layer and a competition layer of processing nodes. All nodes in input layer are connected to every node in the competition layer. Input is fed into the nodes in the input layer, processed through the connections by multiplication with the established connection weights. Then, the sum of incoming value of a node in the competition layer competes with those of neighbor nodes. The competition strategies, which may be different among different CBANN models, are achieved by means of algorithms that assign the winning unit. For example, the winner is the node that has the highest activation value, or the winner is the only unit that is still active after several cycles of inhibition to competition nodes.

The CBANN performs unsupervised learning; a set of training patterns with no output solution is presented to the CBANN and the CBANN finds useful organization of relationship among input patterns. Generally, the CBANN is a self-organizing model that can gradually modify its connection weights based on the characteristics of input patterns. The connection weights can be established through the cycles of competition. CBANNs, nevertheless, employ various details of procedures in updating such connection weights.

In summary, with unsupervised learning and competition approach, the CBANN is best applicable to 1) classification problems where the majority of characteristics of input indicates its class, and 2) optimization problems whose output are not previously known or input–output pairs are not readily available for training ANNs. A resource leveling application falls into the latter case.

The Hopfield network has been known as a pioneer of ANNs for solving a Traveling Salesman Problem (TSP),[a] and has been used in solving several optimization problems. Its architecture consists of highly interconnected simple processing units with symmetric connections where their connection weights are derived from the energy function. The energy function consists of an expression of variables representing characteristics of the problem which constitute 1) the objective of the problem (e.g., the minimum of resource demands), and 2) problem constraints (e.g., sequence of activities). The ANN attempts to find the global minimum of such function. Convergence dynamics associated with the energy minimization process have to satisfy the constraints and the optimization of the objective of the problem.

3. THE DEVELOPMENT OF THE RLANN

While employing the fundamental concept of the CBANN, the RLANN model developed in this paper includes two distinct approaches appropriate to the resource leveling problem. First, the problem is formulated into an energy function as in a Hopfield network (Hopfield & Tank, 1985). The

[a]TSP is a well-known optimization problem where there is a number of cities that a salesman has to visit, and only once. The goal is to find the shortest travel distance.

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

model utilizes an equation of motion containing static connection weights derived from the energy function to determine the activation values for competition of artificial nodes. Rather than being dynamically updated through competition, the values of the weights are generated directly from the information of the problem to be leveled. The benefit is the elimination of training time in an unsupervised learning process. However, the difficulty becomes the generation of a reliable equation of motion whose formulation by theoretical assumption has to be verified by experimental simulations. Second, the strategies used in the competition process are unique. An algorithm is especially designed for resource leveling problems. The development of the model includes proper selection of an ANN scheme, problem formulation, and competition strategies.

3.1. RLANN scheme for representing the problem

A primary step in the development of ANNs for resource leveling is the selection of a scheme appropriate for the problem at hand. This involves three important considerations. First, the scheme must facilitate a construction project schedule for encoding into an input acceptable by the ANN. Second, the scheme should allow output from the artificial nodes to be decoded into a solution of the problem. And third, the scheme must be universal to be applicable to every project schedule without changing the configuration or repeating the training process for the construction of connection weights. An improper ANN scheme might result in a rigid model appropriate for solving only a specific problem.

Figure 1 presents the configuration of RLANN, a two-layered neural network. These layers are input and competition slabs where input is fed into, and where the competition takes place, respectively. Each node in the input layer is fully connected with every processing node in the competition layer. All of these connections, $T_{xi,yj,}$ are feed-forward, except those between nodes that have the same first coordination index in the layers which are then feed-backward connections. For example, $T_{xi,yj}$ are feed-backward when x equals y, where x and i are node indexes in the input layer, and y and j are node indexes in the competition layer.

The input and competition layers consist of node matrices sized $M \times N$ where M is a number of activities and N is a construction project duration, as shown in Figure 2. The

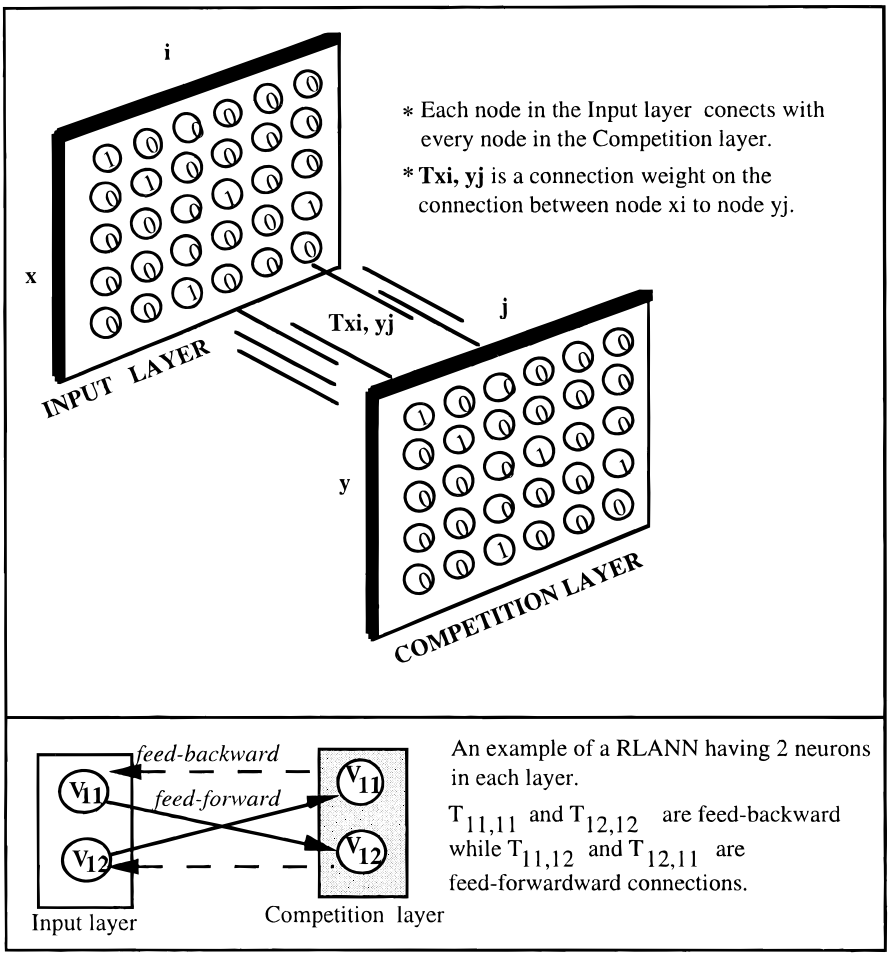


Fig. 1. Two-layered RLANN.

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

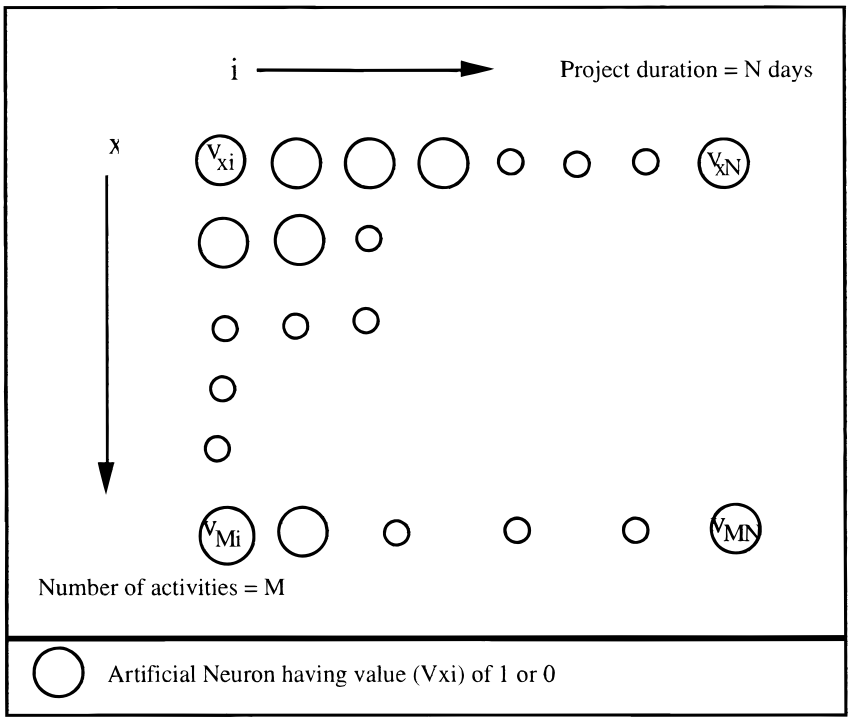


Fig. 2. A structure of neurons representing a schedule.

neural network is developed as a discrete model for which its nodes can present either 1 or 0 value. These analog numbers yield the output of the neural network to be interpreted into a project schedule; they also simplify all schedules to be transformed into 0–1 input format that the neural network can comprehend. Each row of nodes in the matrices represents an activity whereas each column denotes a project day. Regardless of the number of activities and period of project duration, this structure of the nodes is flexible to construction networks of any size.

3.2. Problem formulation

In the case of ANN application for resource leveling, problem formulation has a direct impact on the validity and optimality of a solution. There are several ways to formulate problems onto ANN architecture. These mapping methods are intuitive and can be obtained through both theoretical and experimental research. This paper presents an effective problem formulation based on a project network produced with the critical path method. From the CPM network, the number of critical activities, *m*, project duration, *N*, critical path, and sequences of all activities have been already identified. Each activity *x* has a duration, *dx*, total float, TF_x, early start date, ESD_x, latest finish date, LFD_x, and resource utilization rate, R_x. This information will be used for formulation of the problem and will be input to the neural network as well.

For each activity *x*, the node *xi*, associated with its start date, SD_x, is assigned a value of 1.0 and the rest of the nodes are assigned a value of 0.0; $x = 1, 2, \ldots, M$ and $i = 1, 2, \ldots, N$. V_{xi}, a value of the node *xi*, is expressed as follows:

$$V_{xi} = 1 \quad \text{if activity } x \text{ is started on day } i,$$
$$= 0 \quad \text{otherwise.} \tag{1}$$

To illustrate problem formulation within RLANN architecture, consider a project consisting of three activities with a project duration of 6 days, as shown in Figure 3a. Such schedule can be encoded as input to the proposed RLANN as shown in Figure 3b.

3.3. Energy function

The resource leveling problem is formulated into an energy function indicating how well the resulting resource profile is and if any violations of problem constraints exist. An optimum solution is reached when the energy value is minimum. The energy function consists of two main parts, an objective function and constraints imposed on the problem.

3.3.1. Objective function

Since the objective in resource leveling is to minimize the fluctuation of daily resource usage, the least sum of the squares of daily resource (Burgess & Killebrew, 1962), the minimum moment (Harris, 1978) or other statistical methods can be adopted as a measure of the smoothness of the resource profile. The least sum of the squares is chosen in

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

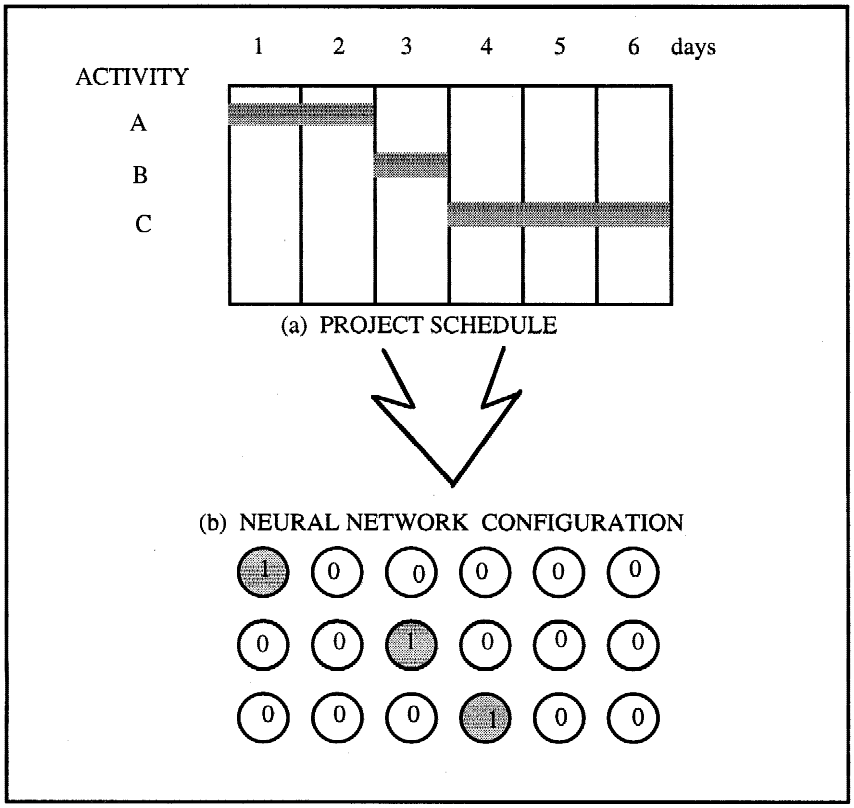


Fig. 3. Formulation problem into the RLANN configuration.

this paper. Therefore, the objective function can be mathematically expressed as follows:

$$\text{Min.} \sum_{i=1}^{N}\left[\sum_{x=1}^{M} R_x \sum_{k=1-d_x+1}^{i} V_{xk}\right]^2. \tag{2}$$

3.3.2. Constraints of the model

The constraints imposed in the resource leveling problem are logical sequences of activities existing in a construction project network and those associated with general assumptions used in traditional resource leveling techniques. For instance, an activity is performed continuously and cannot be divided, and resources required to execute an activity are unlimited and constant during the duration of the activity. In the RLANN, only sequencing constraints are explicitly included in the energy function. The other constraints are implied in the problem formulation.

Based on the equation of motion described later and simulation experiments, the sequencing constraints have to be formulated into both the predecessor relationship, $P(x,x')$, and the successor relationship, $S(x',x)$, where x is a preceding activity and x' is a succeeding activity. P is a set of pairs of activities having a predecessor and successor relationship while S is a set of pairs of activities having a successor and predecessor relationship. These two constraints are essential to force the RLANN to produce a valid schedule. They are formulated as follows:

Precedence constraint.

$$V_{xi}\sum_{i'=1}^{i+d_x-1} V_{x'i'} = 0 \qquad \text{for every pair of activity } x \text{ and } x' \text{ which is a member of set } P. \tag{3}$$

Successor constraint.

$$V_{x'i'}\sum_{i=i'-d_x+1}^{N} V_{xi} = 0 \qquad \text{for every pair of activity } x' \text{ and } x \text{ which is a member of set } S. \tag{4}$$

The objective function and the two constraints combined form the energy function as follows:

$$E = A\left[\sum_{i}^{N}\left[\sum_{x}^{M} R_x \sum_{k=1-d_x+1}^{i} V_{xk}\right]^2\right] + C\left[\sum_{(x,x')\in P}\sum_{i}^{N} V_{xi}\sum_{i'=1}^{i+d_x-1} V_{x'i'}\right]$$

$$+ C\left[\sum_{(x',x)\in S}\sum_{i'}^{N} V_{x'i'}\sum_{i=i'-d_x+1}^{N} V_{xi}\right]. \tag{5}$$

A and C are constants affixed as cost or penalty parameters. When the constraints are violated the cost value appears and causes a high energy value. In a valid schedule, the constraint terms in the energy function become zero; and

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

cost C has no affect. During the competition, if a particular competitive node does not meet the constraints of the problem, it will contain this cost value which leads the node to lose the competition. The value of cost C has to be chosen such that C is much greater than A ($C \gg A$).

3.4. Equation of motion

The equation of motion for the ANN governs alterations of states of the RLANN towards valid and optimum or near-optimum states. After each iteration, every node in the competition layer has its activation value, SUM_{yj}, equal to the sum of the weighted activation of all nodes in the input layer. SUM_{yj} is defined as follows:

$$SUM_{yj} = \sum_{x}^{M} \sum_{i}^{N} (V_{xi} \cdot T_{xi,yj}). \tag{6}$$

In this model, the connection weights, $T_{xi,\ yj}$, are static and derived from the energy function. This concept is encouraged by the initial work of this research which focuses on application of the Hopfield network to resource leveling problems. Although the Hopfield network does not always provide consistency in producing near-optimum schedules, it is found that the static connection weights employed perform well in forcing the state of the system into a valid solution domain. As a result, this sound approach is used together with the designed competition strategies to achieve both valid and optimum or near-optimum solutions in the RLANN. In reference to the discrete model of the Hopfield network (Hopfield, 1982), where the energy function is as in Eq. (7), the energy function in Eq. (5) can be arranged in a manner analogous to Hopfield's energy function as shown in Eq. (8). Then, $T_{xi,yj}$ can be derived as in Eq. (9).

$$E = -\frac{1}{2} \sum_{x=1}^{N} \sum_{\substack{y=1 \\ x \neq y}}^{N} V_x T_{xy} V_y - \sum_{x=1}^{N} V_x I_x, \tag{7}$$

where V_x, V_y, are states of node x and y, respectively, and T_{xy} are connections between nodes x and y, and I_x is a bias value.

$$E = \sum_{x,y}^{M} \sum_{i,j}^{N} V_{xi} \left\{ A \left[\begin{array}{c} R_x R_y \sum_{k}^{N} \theta(k-i)\theta(i-k+d_x-1)\theta(k-j) \\ \theta(j-k+d_y-1) \\ + C[P(x,y)\theta(i+d_x-1-j)] \\ + C[S(x,y)\theta(j-i+d_y-1)] \end{array} \right] \right\} V_{yj} \tag{8}$$

$$\begin{aligned} T_{xi,yj} = A \Big[& R_x R_y \sum_{k}^{N} \theta(k-i)\theta(i-k+d_x-1)\theta(k-j) \\ & \times \theta(j-k+d_y-1) \Big] \\ & + C[P(x,y)\theta(i+d_x-1-j)] \\ & + C[S(x,y)\theta(j-i+d_y-1)], \end{aligned} \tag{9}$$

where $i \neq j$, $x \neq y$,

$$\begin{aligned} \theta(z) &= 1 && \text{if } z \geq 0, \\ &= 0 && \text{otherwise.} \\ P(x,y) &= 1 && \text{if activity } x \text{ is a predecessor of activity } y, \\ &= 0 && \text{otherwise.} \\ S(x,y) &= 1 && \text{if activity } x \text{ is a successor of activity } y, \\ &= 0 && \text{otherwise.} \end{aligned}$$

When $x = y$, $T_{xi,yj}$, are feed-backward connections. Since the feed-backward connections are for updating the activation value of nodes in the input layer following each competition iteration, their established weights are fixed with the value of 1.0 when $i = j$ and 0.0 when $i \neq j$.

3.5. User input to the RLANN

From a CPM project network, find out the number of all activities, critical and noncritical activities, the project and the activities' duration, *ESD*, *LFD* and the resource required for each activity, as well as activity sequences in term of predecessor and successor pairs. The number of activities, M, and project duration, N, are used to form a zero-matrix sized $M \times N$. Organize the activity in increasing order of *LFD*, starting from critical activities followed by noncritical activities (i.e., the activity having the latest finish date will be at the bottom of the list). In the event of a tie, place the activities in an increasing order of Total Float (*TF*). If the activities still tie, place them in the increasing order of number of resources required. Encode the *ESD* of every activity to the value 1.0 and assign it to the matrix at the location associated with the *ESD* of the activity. The finished matrix is an initial input fed to the RLANN.

3.6. Competition strategy in RLANN

While the equation of motion governs activities to be shifted within schedule constraints, the competition process finds the best positions for the activities to achieve the best result. The basic concept of competition employed is as a winner-take-all where each node competes for the best output or low penalty cost to be a winner. In the competition layer, a processing node receives an activation value which is the sum of the weighted signals from all input nodes, excluding input nodes that linked with feed-backward connections. Such an activation value contains two important pieces of information. First, the relative smoothness of the resource profile associated with the considered processing node is given; a low value presents less fluctuation of resource usage. Second, information related to sequencing constraints is given; the activation value is magnificently high due to the cost parameter if the processing node does not meet the problem constraints. The winning node will be the

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

one that has a low activation value (i.e., smoother resource profile and no constraint violation). The winning node is, then, given the new activation value of 1.0 and all other losing units are given the value 0.0. The activation function is a step function and can be mathematically expressed, as in Eq. (10).

$$\begin{aligned} ACTV\,(SUM_{yj}) &= 1 \qquad \text{if } SUM_{yj} > SUM_{yk}, \text{ for all } k,\ k \neq j, \\ &= 0 \qquad \text{otherwise.} \end{aligned} \tag{10}$$

where SUM_{yj} is the weighted sum from Eq. (6) and k is a project day.

In this paper, competition strategies are specifically designed to approach the optimum schedule. First, the competition neighborhood is deliberately selected in each competition iteration. This strategy significantly decreases computation time and unnecessary searching alternatives by narrowing down nodes in the competition to merely those having high potential of being the best choice. For example, at any cycle for a particular activity, the interested neighborhood is the group of nodes in the competition layer corresponding to the activity and its project dates between *ESD* and *LFD*. Second, instead of modifying the connection weights, the activation value of input nodes associated with completed competition nodes are updated through feed-backward connections. Such new activation values will be new states in the next competition iteration. Third, the strategies also include forward and backward competition at which the heavy resource loaded at the late project date can be avoided. In the forward process, competition cycle starts from the activity having the latest *LFD* to the activity having the earliest *LFD* and the nodes attempt to be the winning node with the least cost value positioning the latest possible start date (*SD*). In the backward process, the operation repeats in the opposite direction and the winning node tends to be the costless node representing the earliest possible *SD* of an activity. In the resulting schedule, while keeping the resource profile most leveled, activities are pushed to start at the possible earliest start dates so that their lag time is still available.

Below, the solving steps for a given resource leveling problem employed inside the RLANN will be explained.

3.6.1. *Forward cycle*

Step 1.

- At the competition layer, the competition neighborhood is selected.
- Start from the lowest row of nodes which represents the activities that can start the latest.
- Identify nodes that represent the project days between *ESD* and *LFD* of the activity. Only these nodes will compete with each other.

Step 2.

- At each competition node, its activation value, SUM_{yj}, is calculated from the equation of motion given in Eqs. (6) and (9). Only input nodes connected to the competition node with feed-forward links dispatch their activation values, which are 0.0 or 1.0. The SUM_{yj} is high if the competition node violates the problem constraints or has relatively less potential in becoming a winner to provide an optimum solution.

Step 3.

- The competition commences. As in Eq. (10), a node is assigned as the winner when its activation is lowest or the node has lowest penalty cost. As a winner-take-all, only one node is selected among the competitive neighbor nodes. Hence, in the event of a tie ($SUM_{yj} = SUM_{yk}$), the node that has higher index j or k is chosen. This occasion implies that there is more than one place that the activity can be shifted for smoothness of daily resource usage. In this early cycle, the activity is shifted to start at the latest possible date.
- Update the activation value of the competitive nodes. As in Eq. (10), the selected winner is assigned the value of 1.0 and the rest are assigned the value of 0.0 for their new activation values.

Step 4.

- The input nodes in the input layer corresponding to the completed competition nodes are updated through the feed-backward connections. The new activation value of each node is the sum of weighted activation of such nodes in the competition layer. The other original input with the updated values are then input for the next iteration.

Step 5.

- In the next iteration, the competition is in the next row up from the previous step. The neighborhoods in the competition is based on the *ESD* and *LFD* of the considered activity. Steps 2–5 are then performed. The procedures are repeated by moving the competition field up to the next row until there are no more rows of nodes representing noncritical activities to be determined.

Step 6.

- The RLANN processes Steps 1–5 again and the iterative procedures are terminated when there is no change in output.

3.6.2. *Backward cycle*

Step 7.

- At the competition layer, the competition neighborhood is selected.
- Start from the highest row of nodes representing a noncritical activity.

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

- Identify nodes that represent the project days between the *ESD* and *LFD* of the activity. Only these nodes are in the competition process.

Step 8.

- For each competition node, the activation value, SUM_{yj}, is calculated from the equation of motion given in Eqs. (6) and (9). Only input nodes connected to the competition node with feed-forward links dispatch their activation values.

Step 9.

- Perform the competition. As in Eq. (10), a node is assigned as the winner when its activation is lowest or when the node has lowest penalty cost. Since a winner-take-all strategy is adopted, only one node is selected among the competitive neighbor nodes. Hence, in the event of a tie ($SUM_{yj} = SUM_{yk}$), the node that has lower index, j or k, is chosen. In this cycle, the activity is shifted to start at the earliest possible start date while the smoothness of resource utilization rate is improved

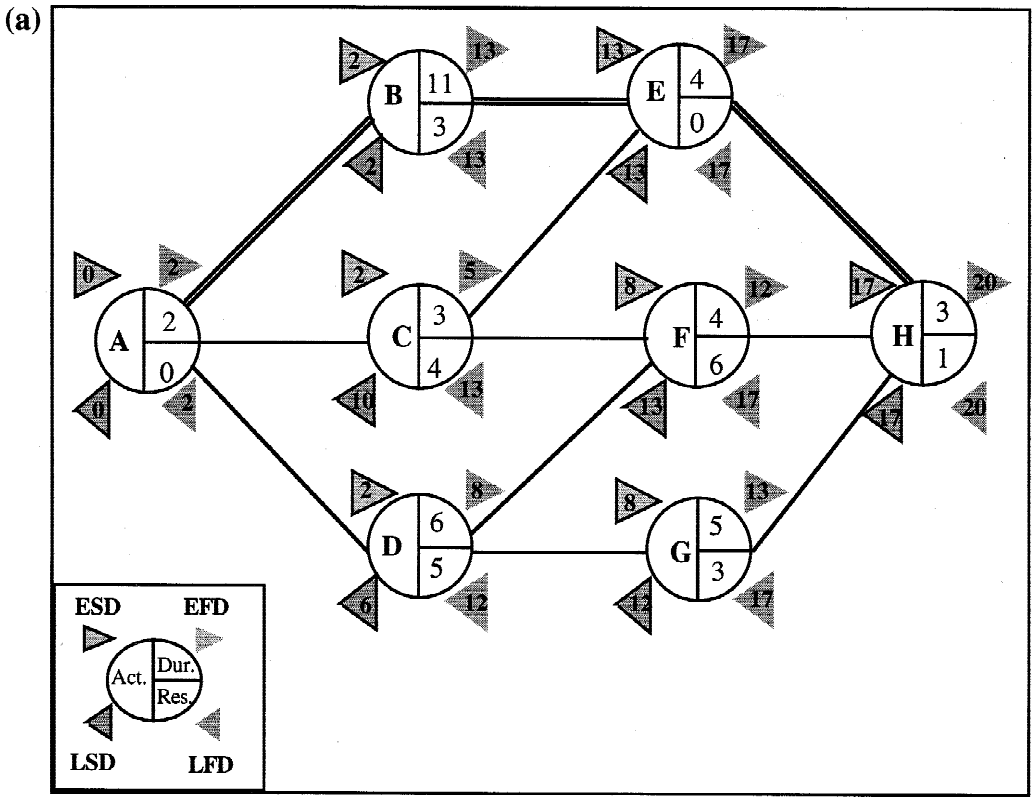


Fig. 4. An example network: (a) precedence diagram, and (b) early start schedule.

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

or maintained. This process provides the activity to be performed with lag times and the resources are distributed among project dates.

- Update the activation value of the competitive nodes. As in Eq. (10), the selected winner is assigned the value of 1.0 and the rest are assigned 0.0 to their new activation values.

Step 10.

- The input nodes in the input layer corresponding to the completed competition nodes are updated through the feed-backward connections. The new activation value of each node is the sum of weighted activation of such nodes in the competition layer. The original input with the updated values is an input for the next iteration.

Step 11.

- In the next iteration, the competition is in the next row lower than that in the previous step. The neighborhoods in competition are based on the *ESD* and *LFD* of the considered activity. Then, Steps 8–11 are performed. The procedures are repeated by moving down to the lower row until the last row of nodes in the competition layer.

Step 12.

- The RLANN processes Steps 7–12 again and the iterative procedures are terminated when there is no change in output.

4. THE USES OF THE RLANN

In this research, the RLANN is developed by using ANSI C language, operating on a serial computer. The RLANN can be performed in various environments, including personal computers, UNIX systems, and parallel machines with some modifications. All input needed for the RLANN is basic scheduling information from a CPM project network. Equation (9) indicates that both predecessor–successor and successor–predecessor pairs are required for the model. However, only the set of activities having a predecessor–successor relationship, $P(x, y)$, is needed for the RLANN since the other set of activity pairs, $S(x, y)$, can be automatically induced from the $P(x, y)$. Although a precedence diagram is easier to encode into an input of the RLANN, an arrow di-

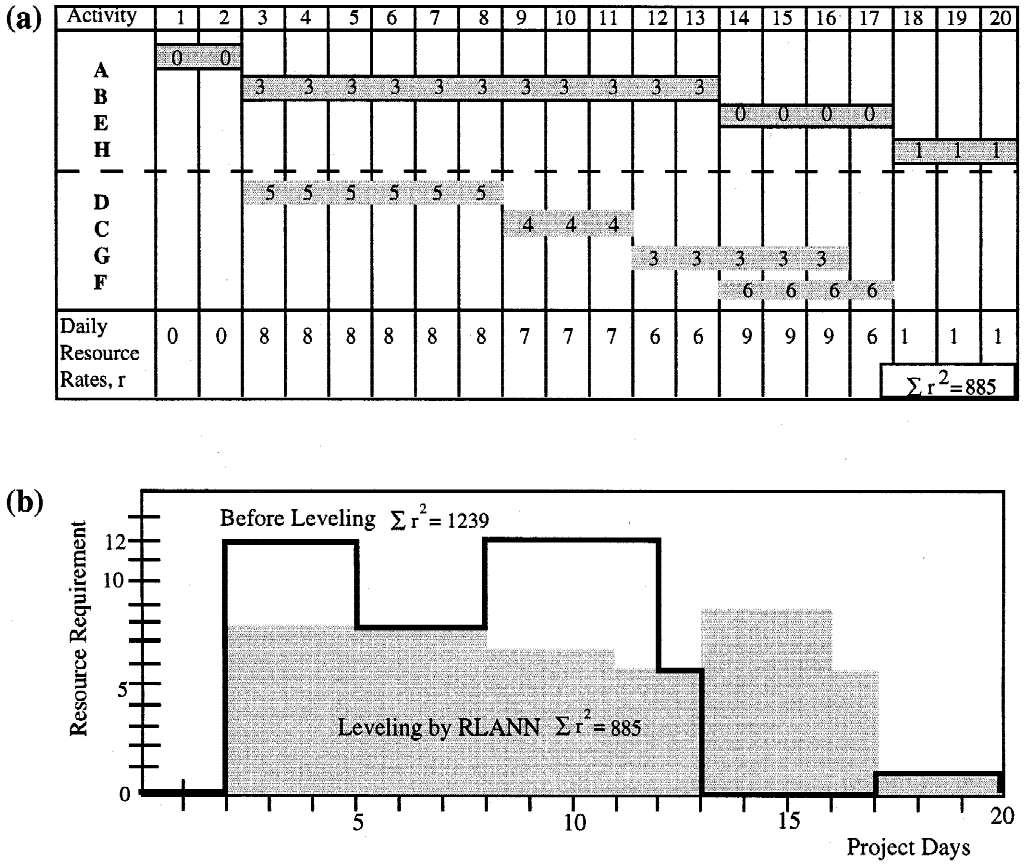


Fig. 5. Resulting schedule and resource histograms. (a) Schedule After Leveling with RLANN (Optimum Solution) (b) Resource Histograms After and Before Leveling by the RLANN.

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

agram is acceptable. In the case of an arrow diagram that has many activities at the start and/or finish node, a dummy activity must be added at the beginning and/or end of the project network. These dummy activities are necessary in the competition process for enforcing activities to start within the project time period.

To demonstrate, the RLANN is applied to level resources in two example project networks. The first example is the application of the RLANN on a precedence diagram. The resulting schedule produced with the RLANN is compared to the schedule generated by the optimization approach using integer-linear programming (Shah et al., 1993). The result is also compared to those solutions provided by three heuristic methods, the least sum of the squares method (Burgess & Killebrew, 1962), the minimum moment method (Harris, 1978), and the modified moment heuristic (Martinez & Ioannou, 1993). The second example illustrates the application of the RLANN on a project network which is presented in the form of an arrow diagram.

4.1. Application on precedence diagram

To compare the resulting RLANN schedule with that of the optimization approach, an example project network is adapted from Shah et al. (1993). Figure 4a illustrates the project CPM precedence diagram which includes eight activities, four critical activities (i.e., A, B, E, and H), and 20 days of project duration. The possible earliest start and late finish dates, activity sequences, and required resource are also shown in Figure 4a. Figure 4b presents the value of the sum of the squares of the daily resource ($\sum r^2$), or the energy value of Eq. (5).

The resulting schedule and the resource histograms before and after leveling are presented in Figures 5a and 5b, respectively. The RLANN produces an optimum schedule with an energy value of 885, improving the early start schedule initiating with an energy value of 1239. The RLANN performs better than the optimization approach using integer-linear programming which produces only a near-optimum schedule with 903 of the sum of the squares of daily resource (see Figs. 6a and 6b).

For this example project, compared with those leveled by the least sum of the squares (Burgess & Killebrew, 1962), the minimum moment (Harris, 1978), and the modified moment heuristic (Martinez & Ioannou, 1993), the solution from RLANN is superior. Figures 7 and 8 represent the schedules and resource profiles of the three heuristic methods, contrast to the optimum histogram produced by the RLANN.

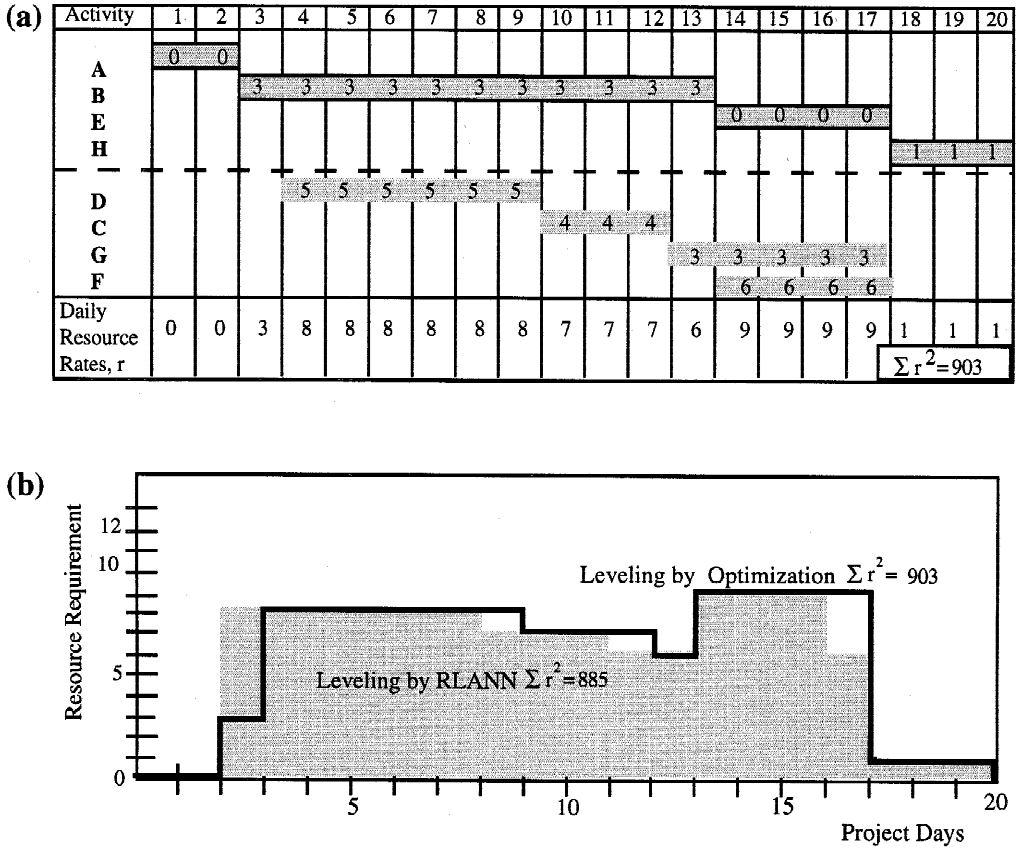


Fig. 6. Comparison of results produced with the RLANN and the optimization approach. (a) Schedule After Leveling with Optimization Approach from Shah et al., 1993 (b) Resource Histograms After Leveling with the RLANN and Optimization Approach (Shah et al., 1993).

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

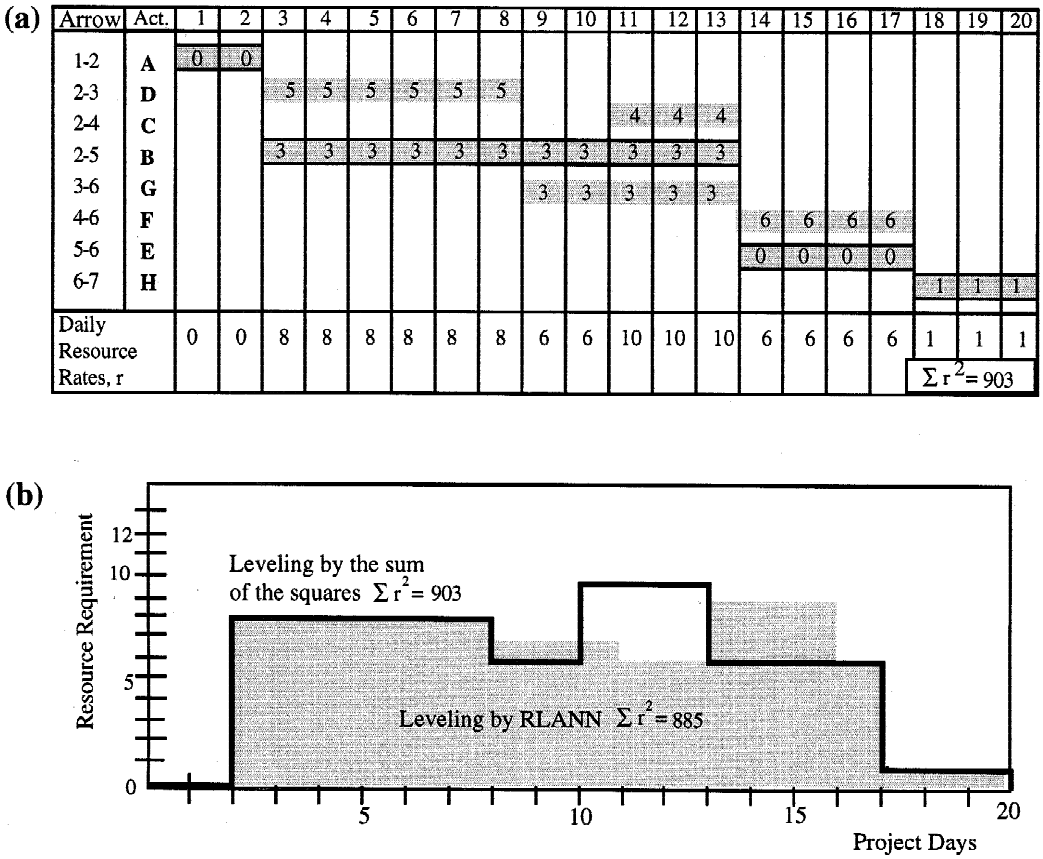


Fig. 7. Comparison of results produced with the RLANN and the sum of the squares method. (a) Schedule After Leveling with the Sum of the Squares (Burgess, 1962) (b) Resource Histograms After Leveling with the RLANN and the Sum of the Squares Method (Burgess, 1962).

In terms of the minimum daily resource requirement, the RLANN produces only nine resources whereas the heuristic methods generate ten.

4.2. Application on arrow diagram

The example arrow diagram and its early start schedule are shown in Figures 9a and 9b, respectively. The project originally consisted of three critical and four noncritical activities. Since there is more than one activity at the beginning and end of the project diagram, dummy activities are placed at the start (i.e., activity 0–1) and the finish (i.e., activity 7–8) of the diagram.

The RLANN levels the fluctuation of resource usage in the early start schedule and produces the optimum schedule with an energy value of 216, as compared to 238 for the early start schedule. Figure 10 shows the optimum schedule which is obtained from the RLANN and the improvement of the resource histograms before and after leveling.

5. EXPERIMENTS AND FINDINGS

The RLANN model has been tested with more than 50 project problems which vary in terms of type (e.g., arrow diagram, precedence diagram, and different kinds of construction projects), size (e.g., number of activities and duration), and complexity (e.g., number of parallel chain of activities, multicritical paths, and multipredecessor sequences). The RLANN is found to provide rapid, consistent, and reliable solutions. For example, the use of the RLANN on a bridge construction project consisting of 100 activities and 250 days of project duration shows that the RLANN levels resources within 11 s on the serial UNIX SUN-OS411B machine. The resulting project schedules are either optimum or near-optimum. Compared to other heuristic or optimization methods, the RLANN is more likely to generate better or, at least, equal results.

These results has also been verified in comparison with standard commercial scheduling programs in all of the project problems tested. For example, the RLANN reduces the fluctuations of daily resource rate of a schedule for a mobile house project by decreasing the sum of the squares value from 375 to 315. On the other hand, commercial scheduling programs, Primavera, Microsoft Project, and Mac Project, produce schedules having sum of the squares values of 331, 337, and 375, respectively.

The task of leveling resource in a real-life construction project, which typically comprises a large number of activ-

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

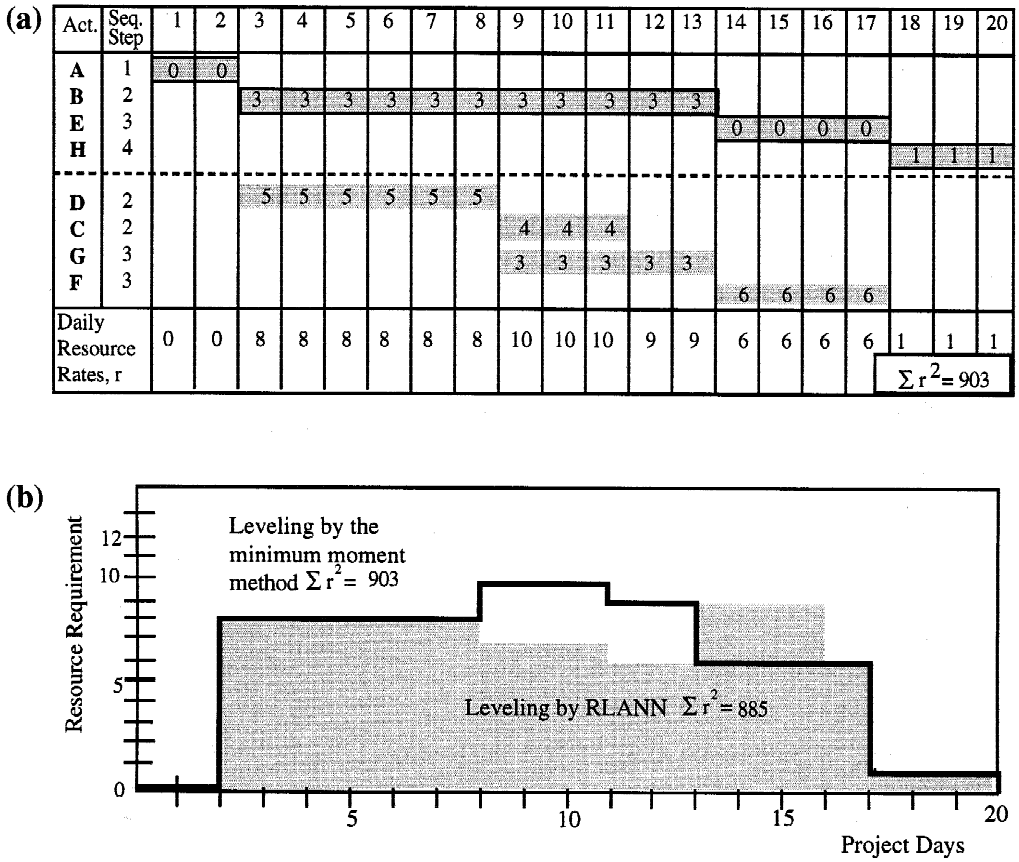


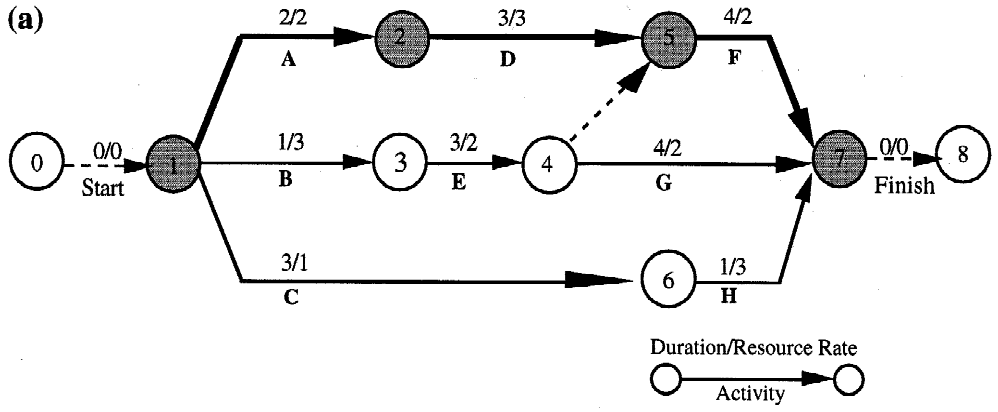
Fig. 8. Comparison of results produced with the RLANN and the minimum moment method (Harris, 1978) or the modified moment heuristic (Martinez, 1993). (a) Schedule After Leveling by the Minimum Moment Method (Harris, 1978) or the Modified Moment Heuristic (Martinez, 1993) (b) Resource Histograms After Leveling with the RLANN and the Minimum Moment Method (Harris, 1978) or the Modified Moment Heuristic (Martinez & Ioannou, 1993).

ities and a high level of complexity, is beyond the capability of the optimization approach due to the limits of current computer technology (Moder et al., 1983; Easa, 1989). Besides the heuristic approach, the RLANN is another new alternative that can be utilized to level resources in this large-scale problem.

Although the heuristic approach is applicable to level resources in such real-life projects, the results demonstrated in this paper confirm that the heuristic approach tends to get trapped at the local minimum, thereby presenting non-optimum results. The RLANN, in contrast, is less susceptible to the local minimum. To illustrate, consider the schedules in Figures 7 and 8. In the Burgess's schedule in Figure 7, due to the heuristics employed, the noncritical activities are forced to start at the latest possible dates that offer less fluctuations of resource usage. As activity shifting progresses, the energy value or the sum of the squares become lower since the shifting step that is accepted is one that decreases in energy value. When there is no improvement on the energy value, the final results is obtained. However, this may or may not be the global minimum that presents the best solution. In this project, for example, the method is stuck in a local minimum where the minimum of resource requirement is still as high as ten. The minimum moment and the modified moment heuristic using improvement factor for selecting priority of the activity to be shifted and a backward process for moving activity backward, also get trapped in this type of problem, as shown in Figure 8. This method is not able to escape from the local minimum because only the back float is considered in the backward process. The RLANN, on the other hand, does not become trapped and shifts activity G by 3 days to overlap with activity F, which is the optimum position, as shown in Figure 5a. The open competition of the neuron enables activities to move forward or backward freely to arrive at the best position in each competition cycle.

On close examination of the RLANN results, in some project networks, the forward and backward cycles employed in the RLANN competition strategies provide different schedules but hold the same sum of the squares of the daily resource usage. The forward cycle produces a possible late start date schedule whereas the backward cycle gen-

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press



(b)

Arrow	Activity	1	2	3	4	5	6	7	8	9
1-2	A	2	2							
1-3	B	3								
3-4	E		2	2	2					
2-5	D			3	3	3				
4-5	dummy									
1-6	C	1	1	1						
4-7	G					2	2	2	2	
5-7	F						2	2	2	2
6-7	H				3					
Daily Resource Rates, r		6	5	6	8	5	4	4	4	2

$\Sigma r^2 = 238$

Fig. 9. Example project network: (a) an arrow diagram, and (b) its early start schedule with the sum of the squares of daily resource.

erates a possible early start date schedule. The former schedule causes resource usage loaded on the later project days and the shape of the resource histogram is more rectangular on the middle to the right side of the histogram. More rectangular shape of a resource histogram indicates less hiring and releasing of resources. For projects of a long duration, this schedule may benefit from avoiding front-end loaded investment, but sacrifices the possibility of postponing the projects' completion if any delay occurs (Easa, 1989). In the latter schedule, resources are distributed among the project dates, and the rectangular histogram is sporadically notched while the low energy value is maintained. This schedule is more practical in construction project in terms of preventing serious project delays since the activities still have lag times to compensate for any delay. The RLANN can provide final solutions by performing either forward/backward cycles, or only forward cycle as specified. Thus, with this capability, the RLANN supplements an early/late schedule analysis to most satisfy those contradicting objectives: front-end loading or late investment.

The main obstacle that may be found in some other ANN applications for solving optimization problems is that the ANN performance is sensitive to penalty parameters and computation time; therefore, the performance is unstable when problems become complicated. Penalty parameters, however, have less influence in the proposed RLANN as long as the value of cost *C* is much greater than the value of *A*. For example, *A* is 1.0 whereas *C* is 1000. A lengthy processing time is common in fully interconnected ANNs when executed on a serial computer. Computation time definitely increases when the problems become larger and more complicated. However, computation time is not a major problem for the RLANN model; this is because of 1) the efficient problem formulation where 1.0 represents only the start date of an activity, 2) the well designed structure of the RLANN where only the involving connection weights are calculated, and 3) the effective competition strategy that selects only high potential nodes to be in competition. Moreover, the RLANN can be implemented on parallel computers, which should enhance the efficiency of the overall system.

6. SUMMARY AND CONCLUSIONS

This paper presents a new robust alternative to traditional resource leveling techniques. The RLANN is described as an optimization tool that can improve construction project

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

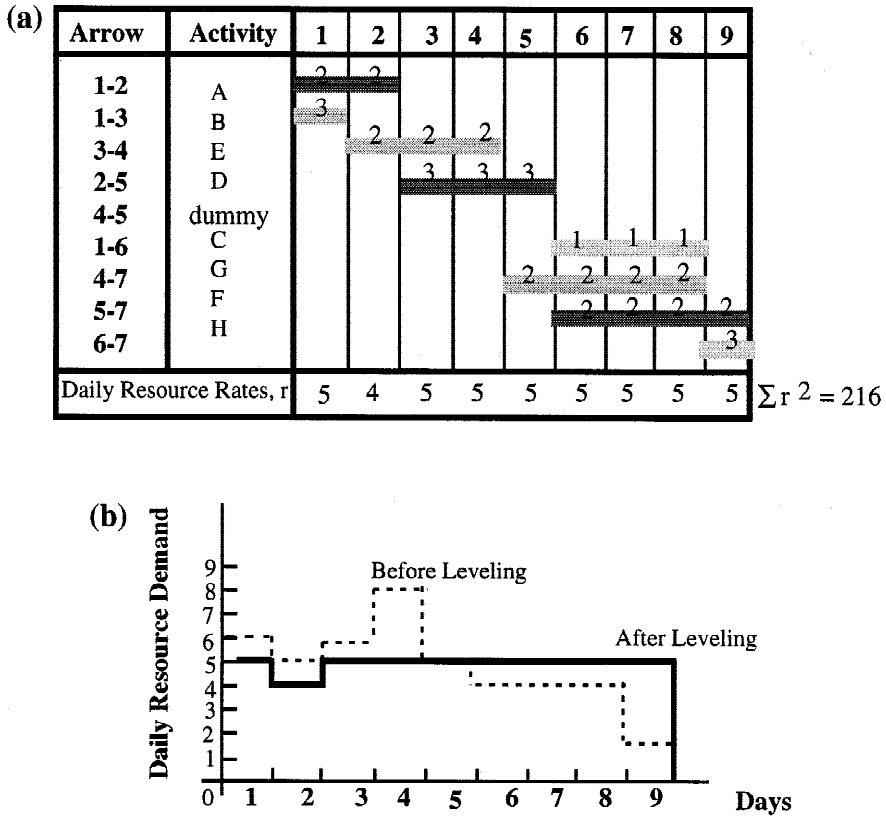


Fig. 10. Resulting schedule and resource histogram before/after leveling by the RLANN.

schedules by producing optimum or near-optimum uses of resources. The RLANN scheme, problem formulations, and solving mechanisms inside the RLANN are explained. The result is then compared to those schedules produced with traditional resource leveling techniques, and confirms that the RLANN is equivalent to, or greater than existing techniques.

The ongoing extension of the RLANN model is the improvement of inside solving mechanism in order to obtain completely independent competition where all competition nodes can compete with each other simultaneously. Simulated annealing (Aarts & Korst, 1989), a stochastic search, may be included in finding the optimum solution region. This will enable the RLANN to produce optimum solutions in more projects. The next extension work will be increasing the RLANN ability to level multiple resources and including reasoning ability by integrating it with a knowledge-based expert system (Kartam et al., 1995) so that the model can deal with restrictions specified in some construction projects.

Furthermore, the concept of the RLANN which has been introduced in this paper can be applied to solve many other optimization problems in civil engineering, for example, bidding evaluation, selection of construction methods and equipment, cost effective design of a project, etc.

REFERENCES

Aarts, E., & Korst, J. (1989). *Simulated Annealing, and Boltzmann Machines*. John Wiley & Sons, New York.

Ahuja, H. (1976). *Project Management: Techniques in Planning and Controlling Construction Projects*. John Wiley & Sons, New York.

Angeniol, B., et al., (1988). Self-organizing feature maps and the traveling salesman problem. *Neural Networks I*, 117–124.

Antill, J., & Woodhead, R. (1982). *Critical Path Methods in Construction Practice*, 3rd Ed. Wiley-Interscience, New York.

Burgess, A., & Killebrew, J. (1962). Variation in activity level in a cyclical arrow diagram. *Journal of Industrial Engineering 13(2)*.

Durbin, F. and Willshaw, D. (1987). An analogue approach to the TSP using an elastic net method. *Nature, 326*, 689–691.

Easa, S. (1989). Resource leveling in construction by optimization. *Journal of Construction Engineering and Management* (ASCE) *115 (2)*, 302–316.

Fang, L., & Li, T. (1990). Design of competitive-based neural networks for combinatorial optimization. *International Journal of Neural Systems 1*, 221–235.

Galbreath, R. (1965). Computer program for leveling resource usage. *Journal of Construction Div.*, (ASCE) *91(1)*, 170–124.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science 11*, 23–63.

Harris R. (1978). *Precedence and Arrow Networking Techniques for Construction*. John Wiley & Sons, New York.

Harris, B. (1990). Packing method for resource leveling. *Journal of Construction Engineering and Management* (ASCE) *116(2)*, 331–350.

Hopfield, J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences (USA), 81*, 3088–3092.

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press

Hopfield, J., & Tank, D. (1985). "Neural" Computation of decisions in optimization problems. *Biological Cybernetics 52*, 141–152.
Kartam, N., Flood, I., & Tongthong, T. (1995). Integrating artificial neural networks and knowledge-based systems for civil engineering. *Journal of Artificial Intelligence in Engineering Design, Analysis and Manufacturing 9(1)*, 13–22.
Kohonen, T. (1988). *Self-Organization and Associative Memory*. Springer Verlag, New York.
Martinez, J., & Ioannou, P. (1993). Resource leveling based on the modified moment heuristic. *Proc. 5th Int. Conf. on Civil and Building Engrg.*, pp. 287–294. ASCE, Anaheim, CA.
Moder, J., Phillips, C., & Davis E. (1983). *Project Management with CPM, PERT, and Precedence Diagramming*, 3rd Ed. Van Nostrand Reinhold Co., New York.
Phillips, D. and Garcia-Diaz, A. (1981). *Fundamentals of Network Analysis*. Prentice-Hall, Englewood Cliff, NJ.
Shaffer, L., Ritter, J., & Mayer, W. (1965). *The Critical Path Method*. McGrawHill Book Co., New York.
Shah, K., Farid, F., & Baugh, J. (1993). Optimal resource leveling using integer-linear programming. *Proc. 5th Int. Conf. on Civil and Building Engrg.*, pp. 501–508. ASCE, New York.

APPENDIX

The following symbols are used in this paper:

$\theta(z)$ = a step function

A = a cost or penalty parameter for an objective function

$ACTV(SUM_{yj})$ = an activation function

C = a cost or penalty parameter for constraints

d_x = duration of activity x

$d_{x'}$ = duration of a successor activity

d_y = duration of activity y

E = an energy function

ESD_x = early start date of activity x

i = an arbitrary project day on input layer

i' = a project day associated with a successor activity

j = an arbitrary project day on competition layer

k = a variable representing project day

LFD_x = latest finish date of activity x

m = number of critical activities

M = number of activities

N = duration of a project in days

$P(x, y)$ = a set of activity pairs having predecessor–successor relation

R_x = resource required to execute activity x

SD_x = a start date of activity x

SUM_{yj} = sum of the weighted activation value at node yj

$S(x, y)$ = a set of activity pairs having successor–predecessor relation

TF_x = total float of activity x

$T_{xi,yj}$ = connection between nodes in input and competition layer

V_{xi} = value of node xi

x = an arbitrary project activity on input layer

X' = a successor activity

y = an arbitrary project activity on competition layer

z = an arbitrary variable

https://doi.org/10.1017/S0890060498123053 Published online by Cambridge University Press