A NEURAL NETWORK APPROACH TO THE SEQUENCING OF CONSTRUCTION TASKS

Ian Flood
Faculty of Architecture and Building

National University of Singapore
10 Kent Ridge Crescent
Singapore 0511

ABSTRACT

The paper describes the development of a neural network based method for solving complex construction operational problems. A brief introduction to neural networks is provided. Particular reference is made to the mechanics and properties of the technique and its potential as a management problem solving tool. Following this, the optimal sequencing of construction tasks (with the objective of minimizing production time) is selected as an exemplary problem for the application of neural networks. A method of tackling this class of problems, using networks developed through a process of simulated evolution, is proposed. The effectiveness of this approach is then evaluated in terms of both the rate at which networks can be evolved and the efficiency of the solutions they produce. The paper concludes with an indication of areas of current research.

1 INTRODUCTION

Construction, among the manufacturing industries, is one of the richest sources of operational problems. The diversity of construction work, the range of methods and resources employed along with their complex interrelationships, and the need to meet tight deadlines and profit margins, all contribute to this fact. Yet, methods of solving all but the most basic of these problems are not generally available. The burgeoning field of artificial intelligence, however, offers much in this direction, providing a number of highly flexible methods of investigating seemingly intractable problems. This paper presents an investigation of the potential of one such technique, that of artificial neural networks, as a means of solving the sequencing problem.

2 NEURAL NETWORKS

2.1 Mechanisms

Neural networks form a class of pattern recognition and classification devices that model, to varying degrees of exactness, the workings of the central nervous system. Despite some initial scepticism [1], they have

become a recent focus of interest, finding potential applications in areas as diverse as speech recognition [2] and the assignment problem [3]. In the following, a brief introduction is given to the principle mechanisms of neural networks as relevant to this paper. A broader introduction to the subject can be found in Rumelhart et al [4].

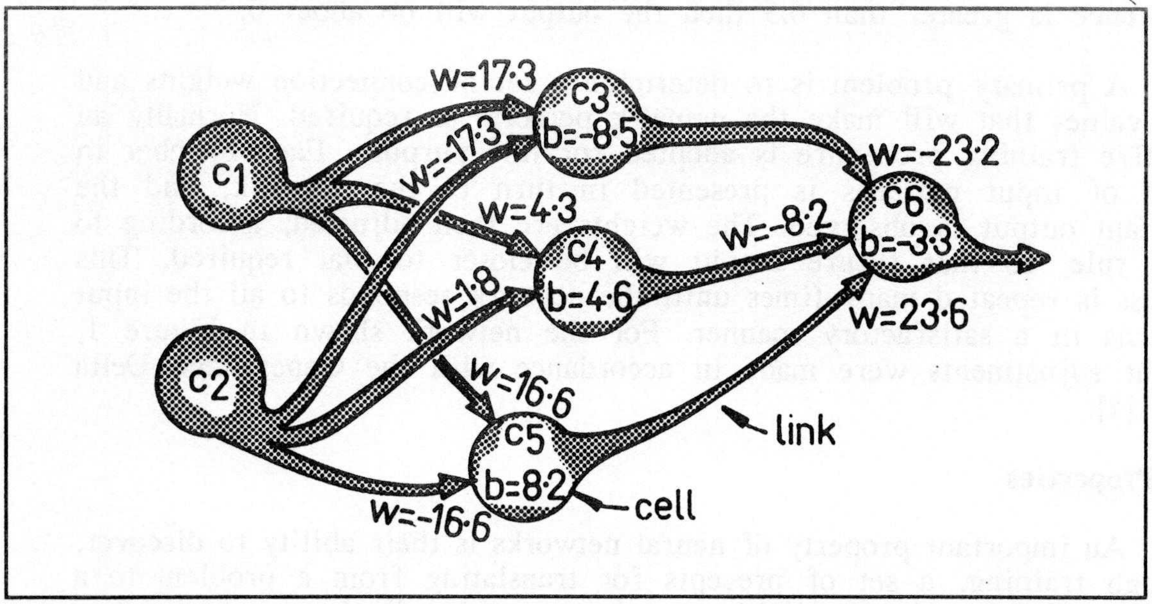

Figure 1 Six-Cell Neural Network

Figure 1 shows an example of a simple network consisting of six interconnected neuron-like cells. Every cell functions in a similar manner, receiving a signal from each of its input links which it then adds to its base value, *b*, and puts through a function (in this case, that shown in Figure 2) to generate a level of activation. The activation level of a cell is then transmitted to its neighbours along its output links. Before this signal is received by a neighbouring cell, it is multiplied by a weighting factor, *w*, at the point of connection between the two cells. The greater the level of activation of a cell, the more it will tend to stimulate (or inhibit, depending on the signs of the connection weights) activity in its neighbours.

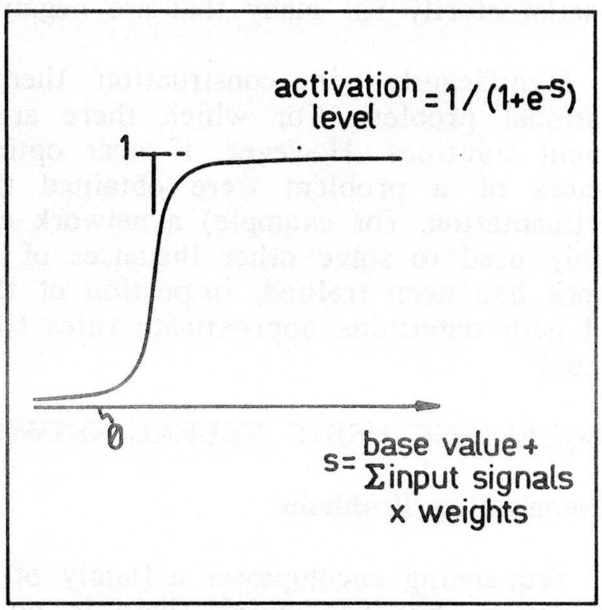

Figure 2 Activation Function

For a given network, it is the set of connection weights and base values that determine its function. In Figure 1, for example, the set of weights and bases result in a circuit that assesses the difference between two values, i_1 and i_2. These values are input as the activation levels of cells c_1 and c_2. If the absolute difference between i_1 and i_2 is less than 0.5 then the output from cell c_6 will be about 1, on the other hand, if the difference is greater than 0.5 then the output will be about 0.

A primary problem is to determine a set of connection weights and base values that will make the network perform as required. Normally an iterative training procedure is adopted for this purpose. Each member in a set of input patterns is presented in turn to the network, and the resultant output is observed. The weights are then adjusted, according to some rule, so that future output will be closer to that required. This process is repeated many times until the network responds to all the input patterns in a satisfactory manner. For the network shown in Figure 1, weight adjustments were made in accordance with the Generalized Delta Rule [4].

2.2 Properties

An important property of neural networks is their ability to discover, through training, a set of precepts for translating from a problem to a solution. These exist implicitly in the networks final set of weights. Moreover, a network can use these precepts to infer solutions to problems beyond those on which it was trained. For example, the network in Figure 1 was taught to assess the difference between just nine example pairs of positive numbers. Yet it works very well for most positive number pairs, and satisfactorily for many that are negative.

Significantly, in construction there are numerous examples of operational problems for which there are no known rules for finding efficient solutions. However, if near optimum solutions to a number of instances of a problem were obtained (from experience or simulation experimentation, for example) a network could be trained with these and possibly used to solve other instances of the problem. Moreover, once a network had been trained, inspection of the sets of weights it developed could help determine approximate rules for solving that type of problem by hand.

3 SEQUENCING USING NEURAL NETWORKS

3.1 Sequencing Problems

Sequencing encompasses a family of problems that occur throughout construction, but for which there is no satisfactory universal solution. These problems are, nevertheless, relatively simple to formulate and in this sense facilitate investigation. For these reasons, sequencing, and more specifically the flow-shop example [5], was selected as a prefatory problem for the application of neural networks.

The flow-shop is one of the most elementary paradigms of the sequencing problem. It consists of a series of processes through which a number of jobs are passed, with the constraints that: a process can operate on just one job at a time; each job must complete a process before moving onto the next; and the order of the processes is the same for all jobs. The time spent at each process varies from job to job. The problem is to determine the sequence of jobs that results in the minimum overall manufacturing time. A practical example is the production of an assortment of precast concrete components, each of which must go through the processes: set-up formwork, fix steel, and place concrete.

3.2 Approach to the Problem

It is possible to conceive of a number of ways of applying neural networks to flow-shop sequencing. The most straightforward of these, and the subject of this paper, is based on the network architecture outlined in Figure 3. Here, the input cells form a grid consisting of one row for each job to be sequenced and one column for each of the processes acting on the jobs. Every input cell forms a connection with every inner cell which, in turn, connect with every output cell. Each output cell represents a job to be sequenced. The idea is that, when a matrix of normalized values, $i_{j,p}$ (representing the various times spent by each job at each process) is presented to the input cells, the network will respond by producing an optimal job sequence across the output cells. The sequence is dictated by the relative levels of activation of the output cells, so that the job represented by the cell with the highest level of activation is first in line.

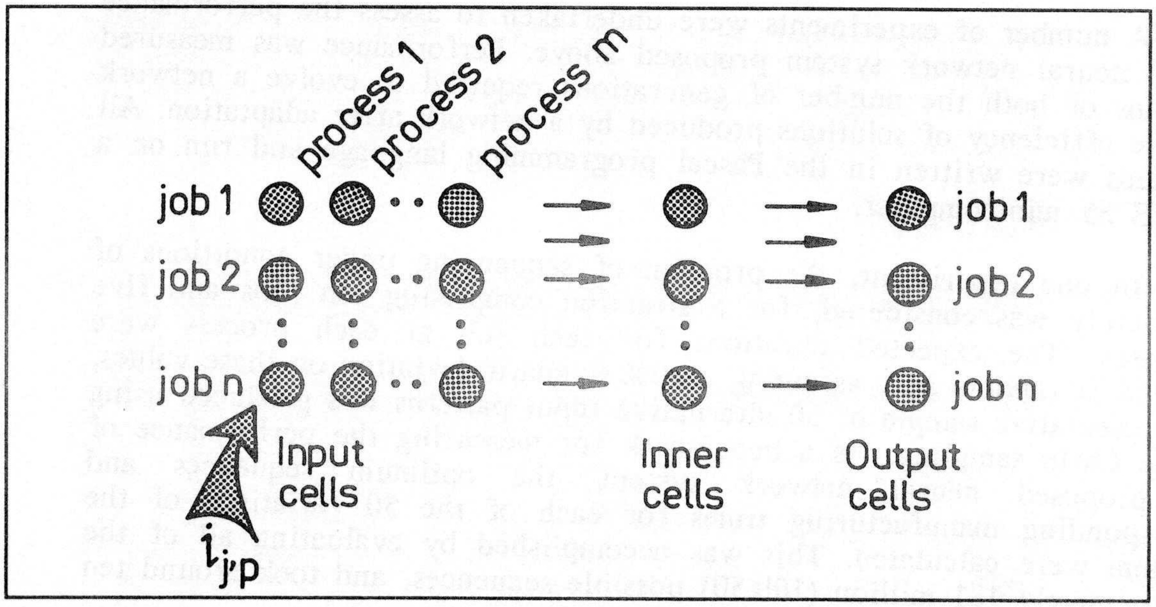

Figure 3 Network used for Sequencing

It is unreasonable to expect that any one network could be used to find solutions to all flow-shop sequencing problems. Such a network would have to be trained to recognize a vast range of problems before it could be relied upon to produce even rough approximations to an optimum solution. Indeed, experimentation along this line has only succeeded in producing a network that works for the most trivial of cases - that of two jobs and two processes. Clearly, the range of possible input patterns has to be limited by confining the application of a network to a specific group of flow-shop problems.

3.3 Network Development Using Simulated Evolution

One outstanding issue is concerned with how to train these networks. The difficulty arises because examples of good solutions to most flow-shop sequencing problems, a prerequisite for training purposes, are not readily available. However, by adopting the technique of simulated evolution [6,7], whereby new networks would be generated from old and accepted/rejected on the basis of the efficiency of the solutions they produce, the difficulty can be circumvented.

Simulated evolution has been found to be an efficient optimization algorithm for a variety of applications [8,9]. Moreover, neural networks appear to lend themselves to adaptation by this method: new generations of networks could be produced from old by swopping groups of cells, mutating connections, and retaining only those that are the fittest for solving the problem at hand.

4 RESULTS AND ANALYSIS

A number of experiments were undertaken to assess the performance of the neural network system proposed above. Performance was measured in terms of both the number of generations required to evolve a network and the efficiency of solutions produced by a network after adaptation. All programs were written in the Pascal programming language and run on a PRIME 55 minicomputer.

In one experiment, the problem of sequencing under conditions of uncertainty was considered, for a situation comprising ten jobs and five processes. The expected durations for each job at each process were selected at random and, assuming a 10% standard deviation on these values, a representative sample of 50 alternative input patterns was produced using Monte Carlo sampling. As a benchmark for measuring the performance of the proposed neural network system, the optimum sequences and corresponding manufacturing times for each of the 50 variations of the problem were calculated. This was accomplished by evaluating all of the approximately 181 million (10!x50) possible sequences, and took around ten days of CPU time.

Trial runs were made evolving a network to solve the 50 sequencing problems. The architecture adopted was based on that shown in Figure 3 with five inner-cells and connection weights all initialized to zero. The simplest of evolutionary operators, that of small random mutations to weight values, was used to produce each new generation. One offspring was produced per generation, and was selected in preference to its parent only if its performance was superior. Performance was measured in terms of the efficiency of the sequences produced by the network. This strategy was found to work most effectively if, first, the number of weights mutated at each generation was kept around one or two and, secondly, weight adjustments were normally distributed with a zero mean and constant standard deviation.

The results from three typical trials are plotted in Figure 4, showing the relative improvement in network performance from generation to generation. All trials reached a stable performance within 200 generations, and required approximately 5 minutes of CPU time to execute. The best network achieved a performance that was just 0.78% away from the optimum compared to an expected 20.0% if sequences were selected purely at random. These results are particularly good in view of the fact that there were usually no more than one to five optimum solutions to each of the 50 sequencing problems.

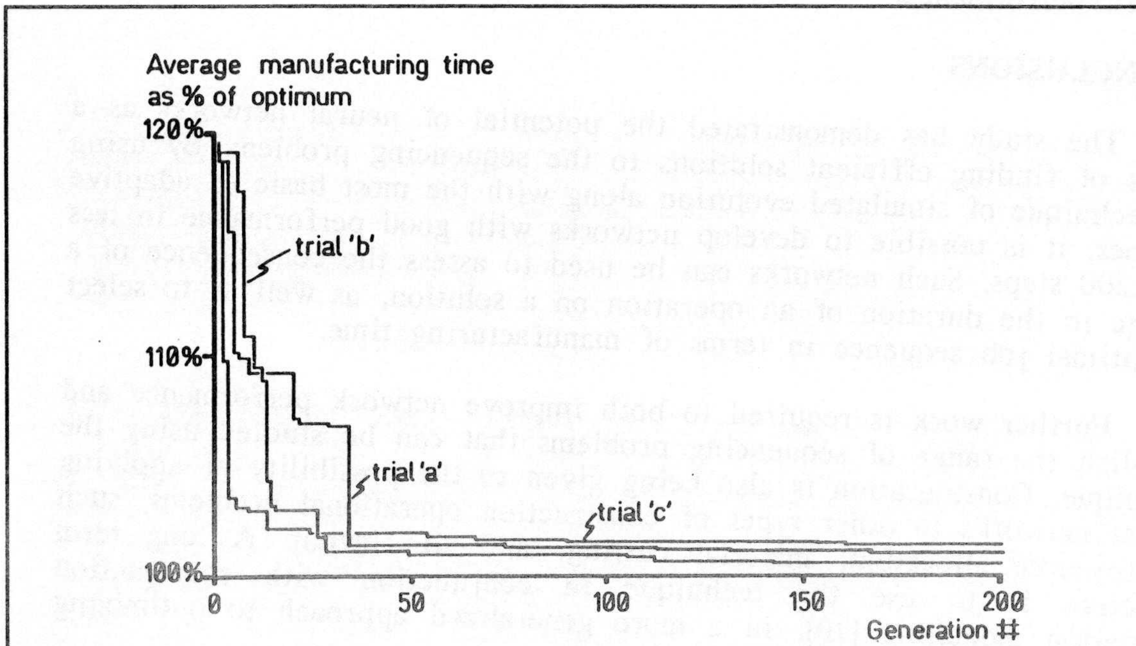

Figure 4 Performance of Network during Simulated Evolution

The network with the best performance was then tested with 50 variations of the sequencing problem that it had not been exposed to during the evolutionary process. In this trial, the network produced solutions that were on average 1.52% from the optimum. This indicates clearly that the network had formed a valid model of the problem. As such, it could be used with confidence to help determine an optimal job sequence. For example, a sensitivity analysis could be performed, ranging the value of each of the inputs to the network. In this way, an identification could be made of the operations most critical in terms of effecting a change in the optimal sequence. Alternatively, the network could be used as a sort of oracle for determining an optimal job sequence under specified operational conditions.

These results are encouraging, though a more rigorous analysis is required, assessing among other factors: performance in relation to higher levels of variance between input patterns; alternative network architectures; and increased numbers of jobs and processes in a problem. However, preliminary experiments have been performed for problems comprising 100 jobs and 10 processes, and results indicate a performance that is characteristic of that attained in the 10 job 5 process example reported above.

Studies are also intended using more sophisticated evolutionary strategies, such as, cross-breeding and the production of more than one offspring per generation, with the aim of attaining greater optimality in network performance.

5 CONCLUSIONS

The study has demonstrated the potential of neural networks as a means of finding efficient solutions to the sequencing problem. By using the technique of simulated evolution along with the most basic of adaptive schemes, it is possible to develop networks with good performance in less than 200 steps. Such networks can be used to assess the consequence of a change in the duration of an operation on a solution, as well as to select an optimal job sequence in terms of manufacturing time.

Further work is required to both improve network performance and establish the range of sequencing problems that can be studied using the technique. Consideration is also being given to the possibility of applying neural networks to other types of construction operational problems, such as, resource allocation, material cutting, and site layout. A long term objective is to use the technique in conjunction with construction simulation modelling [10], in a more generalized approach to optimizing construction activity and resource usage.

REFERENCES

1. Minsky M and Papert S, *Perceptrons*, (MIT Press, 1969).

2. Kohonen T, "The 'Neural' Phonetic Typewriter", *Computer*, (IEEE, March 1988), pp 11-22.

3. Tank D W and Hopfield J J, "Collective Computation in Neuronlike Circuits" *Scientific American*, (December 1987), pp 62-70.

4. Rumelhart D E, et al, *Parallel Distributed Processing*, Volumes 1 and 2, (MIT Press, 1986).

5. French S, *Sequencing and Scheduling*, (Wiley, 1982).

6. Fogel L J, et al, *Artificial Intelligence Through Simulated Evolution*, (Wiley, 1966).

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

8. Kling R, "ESP: A New Standard Cell Placement Package Using Simulated Evolution", Proceedings of the 24th ACM/IEEE Design Automation Conference, (1987), pp 60-66.

9. Irodov V F and Maksimenkov V P, "Application of an Evolutionary Program for Solving the Travelling-Salesman Problem", *Soviet Automatic Control (USA)*, 14, 4, (July-August 1981), pp 7-10.

10. Pilcher R and Flood I, "The Use of Simulation Models in Construction", Proceedings of the Institution of Civil Engineers, Part 1 Design and Construction, 76, pp 635-652, (August 1984).