ELSEVIER

Neurocomputing 8 (1995) 79–91

NEUROCOMPUTING

Scaling properties of neural networks for job-shop scheduling

Simon Y. Foo [a,*], Yoshiyasu Takefuji [b], Harold Szu [c]

[a] *Department of Electrical Engineering, FAMU / FSU College of Engineering, Florida State University, Tallahassee, FL 32316, USA*

[b] *Department of Electrical Enginnering and Applied Physics, Case Western Reserve University, Cleveland, OH 44106, USA*

[c] *Naval Surface Warfare Center, White Oak, R44, Silver Spring, MD 20903-5000, USA*

Received 11 May 1993; accepted 17 January 1994

Abstract

This paper investigates the scaling properties of neural networks for solving job-shop scheduling problems. Specifically, the Tank-Hopfield linear programming network is modified to solve mixed integer linear programming with the addition of step-function amplifiers. Using a linear energy function, our approach avoids the traditional problems associated with most Hopfield networks using quadratic energy functions. Although our approach requires more hardware (in terms of processing elements and resistive interconnects) than a recent approach by Zhou et al. [2], the neurons in the modified Tank-Hopfield network do not perform extensive calculations unlike those described by Zhou et al.

Keywords: Optimization; Scheduling; Mixed integer linear programing; Tank-Hopfield neural networks

1. Introduction

Hopfield-type feedback neural networks have been successfully applied to solving a variety of interesting problems such as the traveling-salesman problem, analog-to-digital conversion [1], resource allocation, tiling problem, k-colorability problem, and many others as described by Takefuji [5]. Recent criticisms of the

* Corresponding author. Email: foo@evax12.eng.fsu.edu

SSDI 0925-2312(94)00011-G

Hopfield networks such as nonconvergence of the network to valid solutions, inability to locate the global minimum, and poor scaling properties, are due to the use of quadratic energy functions as pointed out by Zhou et al. [2]. This work is an extension of our earlier results reported in [4,6] which uses a modified Tank-Hopfield linear programming network with a linear energy function to overcome the shortcomings of the traditional quadratic energy functions.

Job-shop scheduling belongs to the large class of *NP*-complete (nondeterministic polynomial time complete) problems. An NP-complete problem exhibits an exponential growth in the computation time as the size of the problem increases linearly, and is commonly referred to as a 'hard' problem. In general, an NP-complete problem is one which, for some input of size N, takes a time proportional to at least 2^N. NP-complete problems occur naturally in many situations. For example, there are more than $N!$ ways to schedule N courses to various classrooms and instructors, which means that determining a complete schedule of a large number of classes pleasing to both students and professors can become a formidable task!

In general, the job-shop scheduling problem can be stated as follows: Given n jobs that have to be processed on m machines in a prescribed order under certain restrictive assumptions, what is then the optimal order in which each machine handles the jobs? In short, job-shop scheduling is a resource allocation problem. The resources are called *machines* and basic tasks are called *jobs*. Each job may consist of several subtasks referred to as *operations* that are interrelated by precedence restrictions. A close resemblance to the job-shop problem is processor scheduling in a multiprocessor computer system where the problem definition is: Given a deadline and a set of tasks of varying length to be performed on two identical processors, can the tasks be arranged so that the deadline is met?

In a general job-shop problem, each operation is described by a triplet (i, j, k), i.e. operation j of job i is to be executed on machine k. Assuming there are m machines, then each job has exactly m operations with exactly one operation on each machine. If there are n jobs, then each machine must perform n operations, which means that the number of possible sequences is therefore $n!$ for each machine. If the sequences on each machine are independent, there are $(n!)^m$ schedules. However, since each job consists of several operations with a linear precedence structure, many of these schedules will have conflicts and therefore are invalid solutions.

An example 4-job 3-machine job-shop problem is shown by a Gantt chart in Fig. 1(a). Each job-operation-machine triplet (i, j, k) is represented by a block; the length of block is equal to the processing time t_{ijk} required to perform the operation while the numbers on the horizontal axis represent completion times. Note that the processing order of each job by all machines and processing time of each operation are assumed known and fixed (static). A feasible schedule is one where all operations of each job can be placed on one time axis in precedence order and without overlap. In principle, there are infinitely many feasible schedules for a given job-shop problem since an arbitrary amount of idle time can be inserted at any machine between pairs of operations. Superfluous idle time exists in a schedule if an operation can be processed earlier in time without altering the

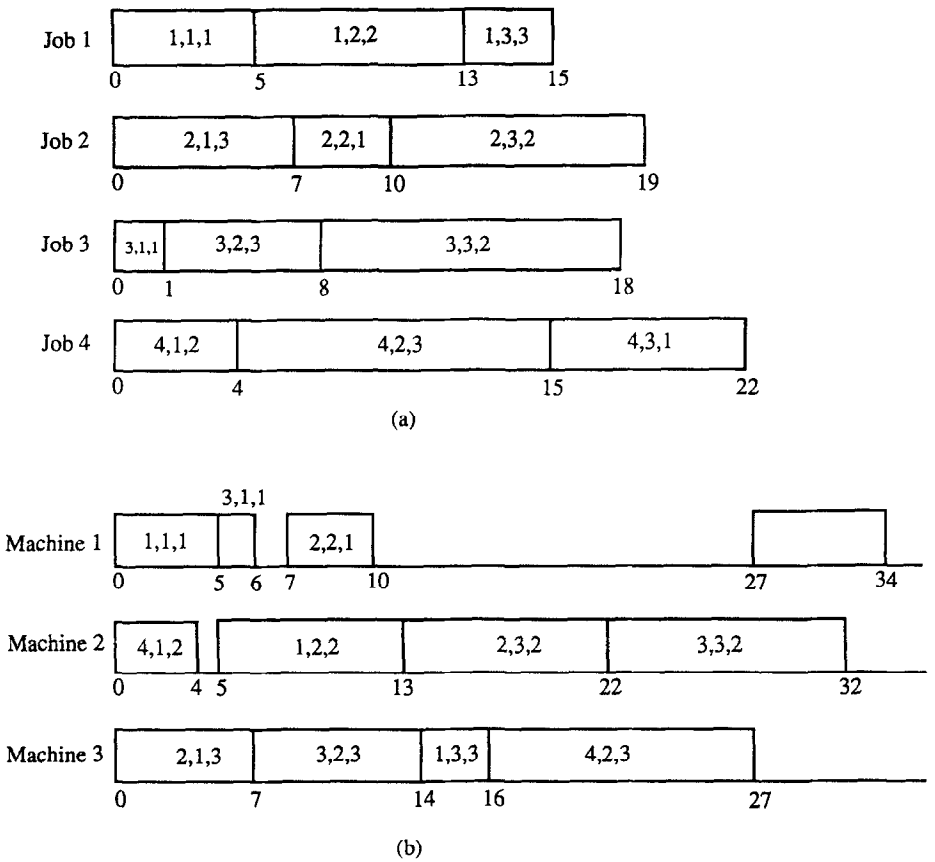


Fig. 1. (a) An example 4-job 3-machine job-shop problem, (the horizontal axis denotes processing times); (b) a near optimum schedule produced by the simulator described in [6].

operation sequences on any machine. In a nondelay schedule, no machine is kept idle at a time when it could begin processing some operation. Although nondelay schedules can be expected to provide very good solutions, there is no guarantee that they provide the optimum solutions. A job-shop problem is completely solved if the starting times of all operations are determined, and the precedence relationships between the operations are not violated. In general, the optimality criteria for machine scheduling can be classified into various groups. These measures of performance include criteria based on completion-dates (i.e. the time at which the last job-operation is completed), flow-times (i.e. the amount of time the job spends in the shop), and makespan (i.e. the total elapsed time required to process all of a given set of jobs). In many cases, previous researchers have used minimization of makespan as the objective function. A near-optimum solution to the 4-job 3-machine problem is shown in Fig. 1(b) using the modified Tank-Hopfield network described in [6].

2. Problem formulation

Several integer linear programming formulations of the machine sequencing problem have been proposed in the past but no attempt has been made to identify and exploit any special structure of this integer linear programming technique. Recently, we [4] proposed a mixed integer linear approach to solving job-shop scheduling. In our approach, the determination of an optimal job-shop schedule can be formulated as a linear programming with integer adjustments to minimize the starting times of all jobs subject to a set of precedence constraints and restrictive assumptions.

Let S_{ik} denote the starting time of job i on machine k, and t_{ijk} the processing time for operation (i, j, k). Assuming operation $(i, j\text{-}1, h)$ precedes (i, j, k), then the inequalities representing precedence constraints in order for a set of S_{ik} to be feasible are:

$$S_{ik} - S_{ih} \geq t_{i,j-1,h} \quad 1 \leq j \leq m, \quad 1 \leq i \leq n \tag{1}$$

where m and n are the number of machines and number of jobs, respectively. The condition that all starting times must be positive results in the constraint:

$$S_{ik} \geq 0 \quad 1 \leq i \leq n \tag{2}$$

It is also necessary to ensure that no two operations are processed simultaneously by the same machine at the same time. For example, if job i precedes job p on machine k (i.e. operation (p, q, k) starts after the completion of (i, j, k)), then

$$S_{pk} - S_{ik} \geq t_{ijk}$$

On the other hand, if job p precedes job i on machine k, it is also necessary that

$$S_{ik} - S_{pk} \geq t_{pqk}$$

Using a 'zero-one' variable y_{ipk} to specify the operation sequence, i.e. $y_{ipk} = 1$ if job i precedes job p on machine k, and $y_{ipk} = 0$ otherwise, the above constraints become:

$$S_{pk} - S_{ik} + H * (1 - y_{ipk}) \geq t_{ijk} \tag{3}$$

$$S_{ik} - S_{pk} + H * y_{ipk} \geq t_{pqk} \tag{4}$$

where constant H represents an arbitrary positive number greater than the maximum value of all processing times t_{ijk}'s such that the constraints (3) and (4) are satisfied.

Therefore, the entire job-shop problem formulation can be summarized as minimizing the cost function

$$\sum_{i=1}^{n} S_{ik_i}$$

subject to

$$
\begin{aligned}
&S_{ik} - S_{ih} \geq t_{i,j\text{-}1,h} && 1 \leq j \leq m, \quad 1 \leq i \leq n \\
&S_{pk} - S_{ik} + H * (1 - y_{ipk}) \geq t_{ijk} && i \geq 1,\ p \leq n, \quad 1 \leq k \leq m \\
&S_{ik} - S_{pk} + H * y_{ipk} \geq t_{pqk} && i \geq 1,\ p \leq n, \quad 1 \leq k \leq m \\
&S_{ik} \geq 0 && 1 \leq i \leq n
\end{aligned}
$$

where k_i is the machine which the last operation of job i is assigned. There are mn constraints of type (1) or type (2), and $mn(n\text{-}1)$ constraints of type (3) or type (4), giving a total number of mn^2 constraints. There are also mn number of S_{ik}'s and $mn(n-1)/2$ number of y_{ipk}'s, resulting a total number of $mn(n+1)/2$ variables. For example, in a 2-job 3-machine problem, there are a total of 12 inequalities with 9 variables in the formulation, i.e.

$$
\begin{aligned}
&S_{11} \geq 0 \\
&S_{12} - S_{11} \geq t_{111} \\
&S_{13} - S_{12} \geq t_{122} \\
&S_{21} - S_{23} \geq t_{213} \\
&S_{22} - S_{21} \geq t_{221} \\
&S_{23} \geq 0 \\
&S_{21} - S_{11} + H * (1 - y_{121}) \geq t_{111} \\
&S_{11} - S_{21} + H * y_{121} \geq t_{221} \\
&S_{22} - S_{12} + H * (1 - y_{122}) \geq t_{122} \\
&S_{12} - S_{22} + H * y_{122} \geq t_{232} \\
&S_{23} - S_{13} + H * (1 - y_{123}) \geq t_{133} \\
&S_{13} - S_{23} + H * y_{123} \geq t_{213}
\end{aligned}
$$

A feasible solution to the above set of contraint equations represents a valid (but not necessary optimal) solution to the job-shop problem.

3. Neural network architecture

The analog Tank and Hopfield linear programming network [1] seeks to minimize a cost function

$$F(\mathbf{A}, \mathbf{V}) = \mathbf{A} \cdot \mathbf{V}$$

where $\mathbf{A}$ is row vector array of N coefficients for N variables which are components of column vector $\mathbf{V}$. This linear cost function is subject to a set of M linear constraints among the N variables:

$$\mathbf{D}_j \cdot \mathbf{V} \geq \mathbf{B}_j \quad j = 1, \ldots, M$$

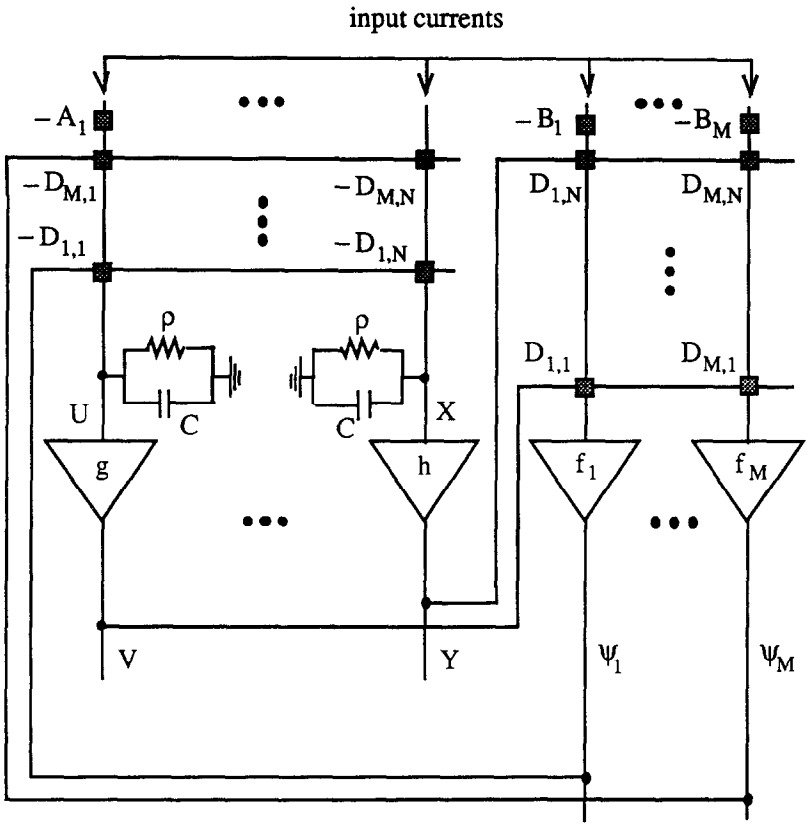


Fig. 2. The modified Tank and Hopfield network consisting of linear and nonlinear processors for solving job-shop scheduling problems, see [6].

where matrix $\mathbf{D}_j$ contain the N variable coefficients of constraint equation j and column vector $\mathbf{B}_j$ are the bounds.

Our proposed network for solving mixed integer-linear programming problems is based on a slightly modified Tank and Hopfield network, as shown in Fig. 2. The difference is the addition of nonlinear step-function h-amplifiers. The output voltage V_i of linear g-amplifier represents the starting time of each operation after minimization, while output Y_i of nonlinear h-amplifier represents the zero-one variable. The components of $\mathbf{A}$ are proportional to the input currents fed into the g- and h-amplifiers. Each g- and h-amplifier has an input time constant $\rho_i C_i$ which acts as a memory element. The descriptions of analog components for implementing the modified neural network can be found in [3] and [6].

The M outputs (ψ_j) of the f-amplifers represent constraint satisfaction. This set of amplifiers have a nonlinear transfer function

$$\psi_j = f(U_j) = \begin{cases} 0 & U_j \geq 0 \\ -U_j & U_j < 0 \end{cases}$$

where

$$\mathbf{U}_j = \mathbf{D}_j \cdot \mathbf{V} - \mathbf{B}_j$$

i.e. if $U_j < 0$ the output ψ_j of the f-amplifier will be a large positive value indicating a violation of the jth constraint. When this happens, a current of value $(\psi_j D_{ji})$ will be fed back into the inputs of the g- and h-amplifiers, causing the outputs V_i to oscillate between $+\text{Vcc}$ and $-\text{Vcc}$, where Vcc is the power supply voltage of the operational amplifiers. If $U_j \geq 0$ the output ψ_j of the f-amplifier is zero corresponding to a valid solution and no current will be fed back to the g- and h-amplifiers. As a result, the output voltages of the g- and h-amplifers will be stable and directly proportional to the input bias currents A_i's.

Since the currents generated by the g- and h-amplifiers are summed at the inputs of the f-amplifiers, it is necessary that the time constants $(\rho_i C_i)$ at the inputs of g- and h-amplifiers be matched. If we assume the response time of the f-amplifier is negligible compared to the g- and h-amplifiers, then the dynamics of the g-amplifier can be described by:

$$C_i \frac{\mathrm{d}U_i}{\mathrm{d}t} = -A_i - \frac{U_i}{R_i} - \sum_j^M D_{ji} f(D_j \cdot V - B_j) \tag{5}$$

where $M = mn^2$ (i.e. total number of constraints). Similarly, the differential equation describing the behavior of the h-amplifier is:

$$C_i \frac{\mathrm{d}X_i}{\mathrm{d}t} = -A_i - \frac{X_i}{R_i} - \sum_j^M D_{ji} f(D_j \cdot Y - B_j) \tag{6}$$

Consider an energy function for the entire circuit of the form:

$$E = \sum_i^P A_i V_i + \sum_i^P \frac{1}{R_i} \int_0^{V_i} g^{-1}(V)\, \mathrm{d}V + \sum_i^P \sum_j^M \int f(z)\, \mathrm{d}z$$

$$+ \sum_i^Q \frac{1}{R_i} \int_0^{Y_i} h^{-1}(Y)\, \mathrm{d}Y + \sum_i^Q \sum_j^M \int f(w)\, \mathrm{d}w$$

where

$$U = g^{-1}(V)$$

$$X = h^{-1}(Y)$$

$$z = D_j \cdot V - B_j$$

$$w = D_j \cdot Y - B_j$$

$P = mn$ (i.e. total number of S_{ik}'s)

$Q = mn(n-1)/2$ (i.e. total number of y_{ipk}'s)

It can be easily shown that E is always a decreasing energy function which seeks out a minima in E and stops. The proof is as follows:

$$\begin{aligned}
\frac{dE}{dt} &= \sum_i^P A_i \frac{dV_i}{dt} + \sum_i^P \frac{1}{R_i} U_i \frac{dV_i}{dt} + \sum_i^P \sum_j^M f(z) \frac{dz}{dt} \\
&\quad + \sum_i^Q \frac{1}{R_i} X_i \frac{dY_i}{dt} + \sum_i^Q \sum_j^M f(w) \frac{dw}{dt} \\
&= \sum_i^P A_i \frac{dV_i}{dt} + \sum_i^P \frac{U_i}{R_i} \frac{dV_i}{dt} + \sum_i^P \sum_j^M f(D_j \cdot V - B_j) D_{ji} \frac{dV_i}{dt} \\
&\quad + \sum_i^Q \frac{X_i}{R_i} \frac{dY_i}{dt} + \sum_i^Q \sum_j^M f(D_j \cdot Y - B_j) D_{ji} \frac{dY_i}{dt} \\
&= \sum_i^P \frac{dV_i}{dt} \left\{ -C_i \frac{dU_i}{dt} \right\} + \sum_i^Q \frac{dY_i}{dt} \left\{ -C_i \frac{dX_i}{dt} \right\} \\
&= -\sum_i^P C_i g^{-1'}(V_i) \left(\frac{dV_i}{dt} \right)^2 - \sum_i^Q C_i h^{-1'}(Y_i) \left(\frac{dY_i}{dt} \right)^2
\end{aligned}$$

Since C_i is positive, and $g^{-1'}(V_i)$ and $h^{-1'}(Y_i)$ are monotone increasing and step functions, respectively, it follows that $dE/dt \leq 0$ with $dE/dt = 0$ if $dV_i/dt = dY_i/dt = 0$ for all i. In other words, the time evolution of the network is a motion in state space which seeks out a minima in E and stops.

4. Scaling properties

The total number of constraints is mn^2 which is also equal to the number of f-amplifiers required. The total number of variables is $mn(n+1)/2$ which is also equal to the number of g- and h-amplifiers. Fig. 3 shows the total number of constraints/variables required to solve a job-shop problem for a fixed $m = 100$ and varying n and a fixed $n = 100$ and varying m, respectively. For a fixed $m = 100$ machines problem, the number of constraints/variables show a polynomial growth for practical n number of jobs. However, for a fixed $n = 100$ jobs, a linear relationship exists between the number of constraints/variables and the number of machines, m.

The total number of amplifiers (neurons) required to implement the network is simply the sum of the number of constraints and variables in a given job-shop problem, as illustrated in Fig. 4. For a fixed $n = 100$ jobs, the total number of amplifiers (i.e. f, g, and h types) required grows linearly as the number of operations assigned to the machines; while a fixed $m = 100$ machines allocation shows an polynomial growth as the number of jobs increases.

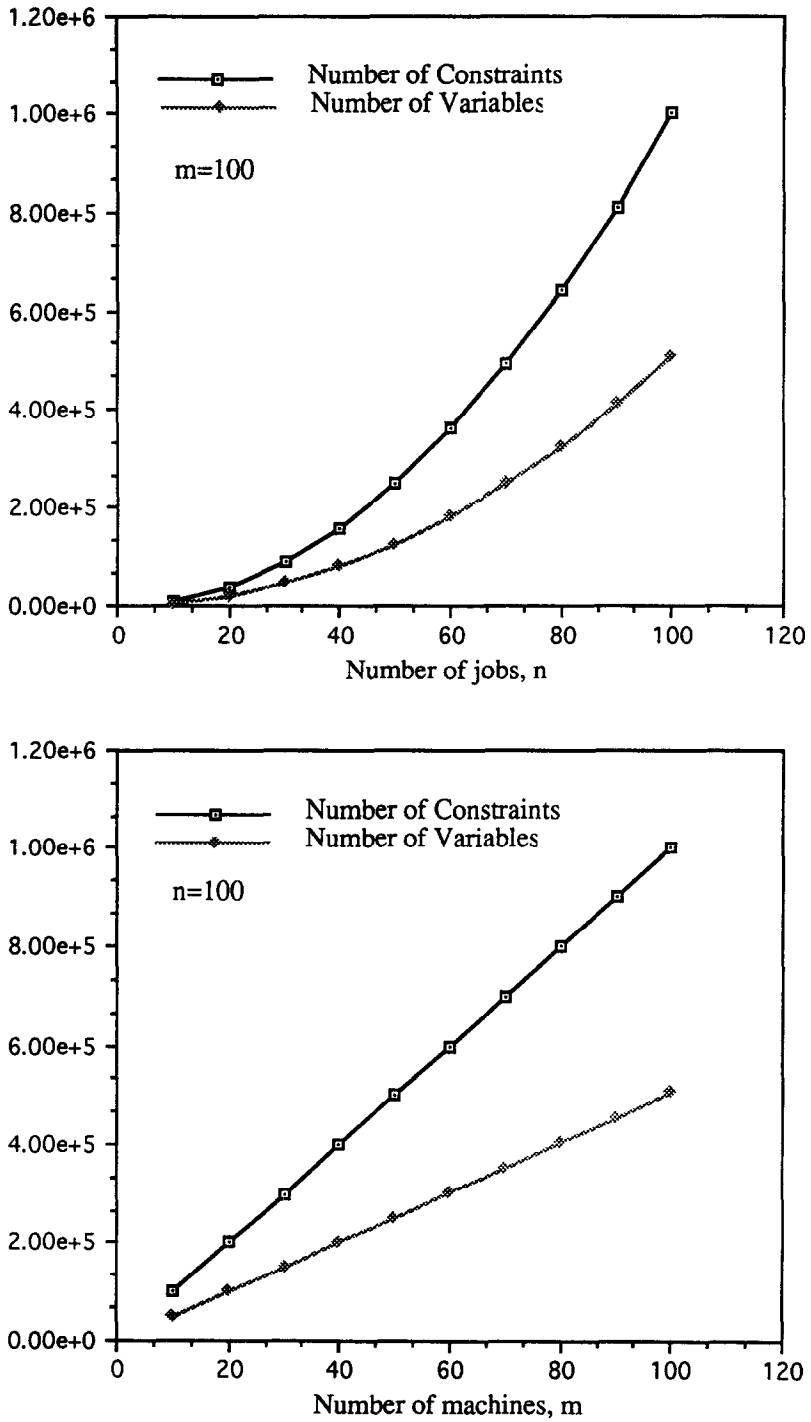


Fig. 3. Graphs of total number of contraints and variables required for a n-job m-machine job-shop problem.

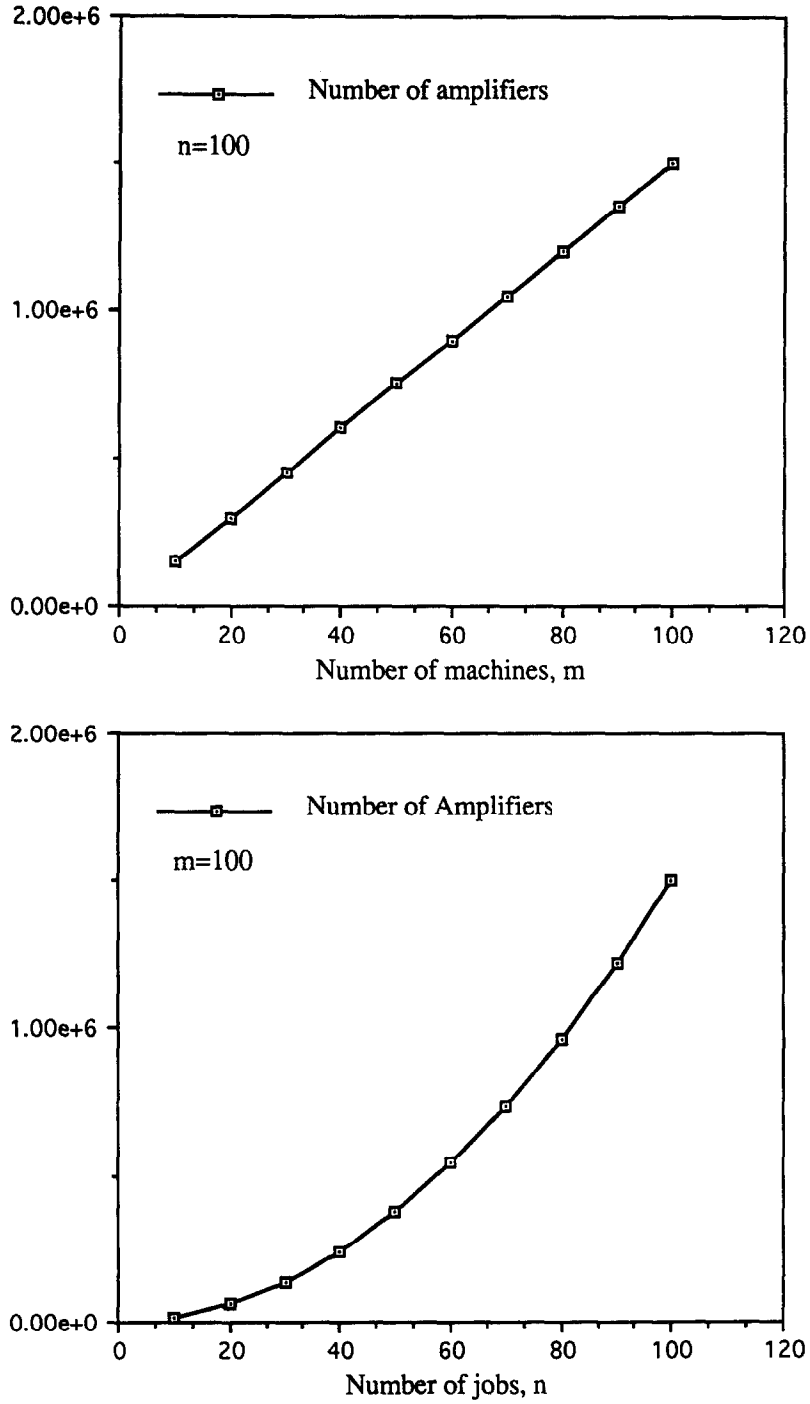


Fig. 4. Total number of amplifiers (f, g, and h-types) required for a n-job m-machine problem.

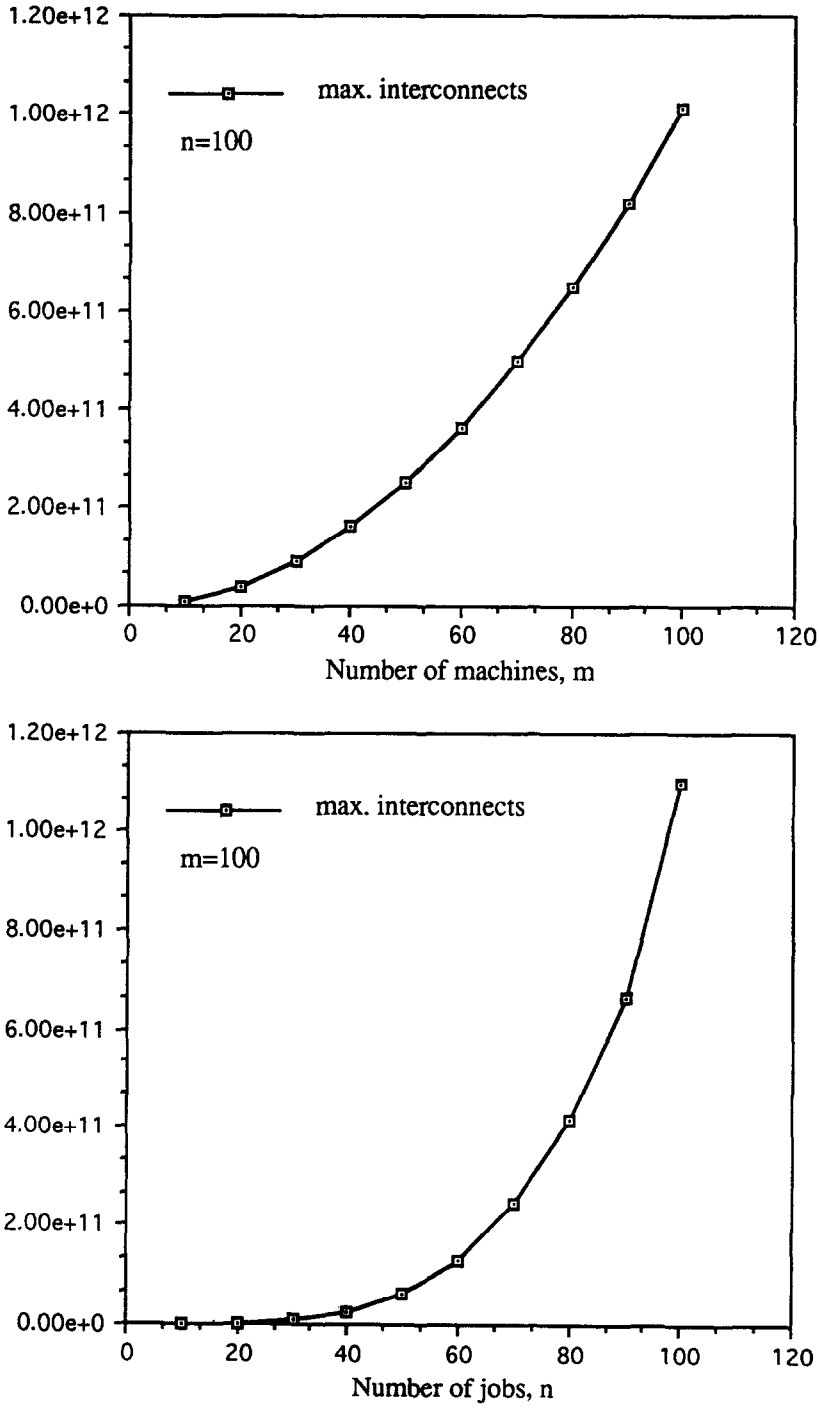


Fig. 5. Total maximum number of interconnects required for a n-job m-machine problem.

The total maximum number of interconnects required is two times the product of the number of constraints and the number of variables, as observed from the architecture of Fig. 2. However, in practice, only a small fraction (about one-fourth) of the interconnect matrix is actually utilized (see [6]). Fig. 5 shows the maximum interconnects required for practical problems. Even though these numbers seem astronomical, recent and future advances in optical and VLSI technologies would make the hardware implementations feasible.

5. Discussions

We have shown how a modified Tank and Hopfield network can be utilized to solve difficult optimization problems such as the classical job-shop scheduling. Zhou et al. [2] also presented a modified Tank and Hopfield analog computational network which exhibits linear scaling property, and thus appears to be better than the original approach by Foo and Takefuji [4]. However, a closer look reveals that their approach requires extensive computations by each neural processor. Thus, there is less network complexity at the expense of more complex processing by each neuron. This contradicts the popular concept of neurons with simple activation functions. Our approach may require more neurons and interconnects, but each neuron has a very simple activation function. Thus, our network is self-contained and does not need extensive calculations as required in the approach by Zhou et al.

Acknowledgements

The authors would like to thank Lisa R. Anderson and Joe Cordaro for implementing the linear programming circuit which was later modified to solve the job-shop scheduling.

References

[1] D.W. Tank and J.J. Hopfield, Simple 'neural' optimization networks: An A/D Converter, signal decision circuit, and a linear programming circuit, *IEEE Trans. Circuits Syst.* CAS-33 (5) (May 1986) 533–541.

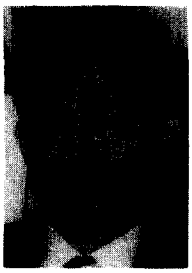
[2] D.N. Zhou, V. Cherkassky, T.R. Baldwin and D.E. Olson, A neural network approach to job-shop scheduling, *IEEE Trans. Neural Networks* 2 (1) (Jan. 1991) 175–179.

[3] S.Y. Foo, L.R. Anderson and Y. Takefuji, Analog components for the VLSI of neural networks, *IEEE Circuits Devices* 6, (4) (July 1990) 18–26.

[4] S.Y. Foo and Y. Takefuji, Integer-linear programming neural networks for job-shop scheduling, *Proc IEEE IJCNN '88*, San Diego, CA (1988) 341–348.

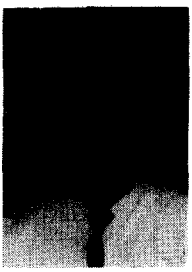
[5] Y. Takefuji, *Neural Network Parallel Computing*, (Kluwer, Dordrecht, 1992).

[6] S.Y. Foo, Y. Takefuji and H. Szu, Job-shop scheduling based on modified Tank-Hopfield linear programming networks *Eng. Appl. Artificial Intelligence* (3) (April/May 1994).



Harold H. Szu received a Ph. D. from Rockefeller University in 1971, and has since worked at Naval Research Laboratory over fifteen years. He has published more than 150 technical papers, and holds 6 patents. He is a member of Institute of Advanced Study at Princeton, a senior member of IEEE, a Fellow of SPIE. He hosted the Advanced Optical Technologies in Leesburg, Virginia, in 1986, and the book *Optical and Hybrid Computing* derived from that meeting and the formation of International Neural Network Society (INNS) in the following year. He is a cofounder of the INNS, has been its Secretary and Treasurer since its inception in 1987, and served as INNS's President in 1993. He has since edited two special issues in Optical Engineering: Sept, 92 *Wavelet Transform* and July 94 *Adaptive Wavelet Transform*. He was the editor-in-chief of *Journal of Neural Network Computing* 1989–1991. Dr. Szu is currently the information science group leader at Naval Surface Warfare Center. His current research involves neural network pattern recognition, adaptive wavelet preprocessing, optical computing, photonic storage, and molecular computing.

Yoshiyasu Takefuji is a tenured associate professor on faculty of environmental information at Keio University since April 1992 and also on faculty of Electrical Engineering at Case Western Reserve University since 1988. Before joining Case, he taught at the University of South Florida for two years and the University of South Carolina for three years. He received his BS (1978), MS (1980), and Ph.D. (1983) from Electrical Engineering from Keio University under the supervision of Professor Hideo Aiso. His research interests focus on neural network parallel computing for solving real-world problems. He is also interested in VLSI applications and silicon architecture. He authored a book entitled *Neural Network Parallel Computing*, from Kluwer Publishers in 1992 and coauthored three books *Digital Circuits* (Ohm-Sha Publishers) in 1984 and *Neurocomputing* (Baifukan Publishers) in 1992 and *Neural networks in design and manufacturing* (World Scientific Publishers) in 1993.



Simon Y. Foo received his BSEE, MSEE, and Ph.D in Electrical Engineering from The University of South Carolina in 1983, 1984, and 1988, respectively. Dr. Foo is currently an assistant professor in the Department of Electrical Engineering, FAMU/FSU College of Engineering, The Florida State University, Tallahassee, Florida. His research interests include analog and digital VLSI design and test, microelectronics, and neural networks. He has published more than 16 technical papers in the area of neural networks and VLSI CAD, and contributed to two books.