A Framework for Embedded Hypercube Interconnection Networks: Based on Neural Network Approach

Mohd. Kalamuddin Ahmad
Mewar, University
Chhittorghara, India

Mohd. Husain
U.P.Tech., University Lucknow,
Lucknow, India

A.A. Zilli
Integral University,
Lucknow, India

ABSTRACT
This paper is concerned with routing of data in an embedded hypercube interconnection using the approach based on neural net architecture. To present a framework of the interconnection network consist number of nodes and number of connections. In this paper we first show that n dimensional hypercube can be embedded in layer neural layer network such that for any node of hypercube, its neighboring nodes of other layer are evenly partition into layers where each layer shares a manipulating or resulting data of different layers. Under this embedding network to fixed target and varying data input to produce output of the two incidence matrix of k-ary n-cube network to embedded in architecture.

Keywords
Hypercube network, Parameters, Embedded network, Layered network, Mean square error, Performance, Validation, Scalability, MIPS, Neurons.

1. INTRODUCTION
Hypercube are a powerful connection network that is able to perform various kinds of parallel computing and simulate many other networks. Hypercube have been widely studied in interconnection networks [2], [4], [6], [7]. A number of other topologies, such as paths, trees, rings, mesh torus can be embedded into hypercube. There are also many related results in hypercube with faulty vertices or links or connection one of the most popular hypercube, which can be the extension of hypercube and can be constructed by adding a links to every pair of nodes with address. This type of embedded hypercube has been improving the SIMD system's performance over a regular hypercube [8] [14] [15] [16].

1.1 Related Works
Parallel computers with mesh and torus topologies have been very successful in offering high computational power together with high speed processor interconnections. Past research on parallel computing has resulted in a well established model for computing on lower dimensional meshes and tori [1], [3], [5], [8]. A large scale up to designed and analyzed within that model can be found in the literature. Unfortunately, most of these topology of network lack the simplicity of their designed for the hypercube models.

The mesh network computation model is quite hard to work with, and probably for this reason several architects of massively parallel machines rely on more powerful networks like hypercube, trees for the implementation of mesh and hypercube.

Solve a given problem more rapidly, or to enable the solution of a problem that would otherwise be impracticable by a single worker. The way the nodes are connected to one another various among machines. In direct network architecture, each node has a point to point, or direct, connection to some number of nodes, called the neighboring nodes. Direct network have become popular architecture for constructing massively parallel computers because they scale well, that is the number of nodes in the system increases as well as communication bandwidth and processing capability of the system also increase.

In direct network architecture, each node has a *point-to-point*, or direct, connection to some number of other nodes, called neighboring nodes. Neighboring nodes may send packets to one another directly, while nodes that are not directly connected must rely on other nodes in the network to transfer packets from source to destination. Although a router's function could be performed by the corresponding local processor, dedicated routers are used to allow overlapped.

Computation as well as communication within each node, router supports some number of input and output channels. Internal channels connect the local processor memory to the router. External channels are used for communication between routers, and, therefore nodes. By connecting the input channels of one node to the output channels of other nodes, the *topology* of the direct network will be defined. For topologies in which packets may have to traverse some intermediate nodes, the *routing algorithm* determines the path selected by a packet to reach its destination. At each node, the routing algorithm indicates the next channel to be used. Efficient routing is critical to the performance of interconnection network [1], [6], [7].

1.2 Problem definition & Scope Issues
Interconnection network is a key component of a parallel computer. Many issues need to be considered when designing on a specific topology for connecting a set of processors. Given the rapid technological advances in VLSI, it is reasonable to conceive of a huge number of processors being integrated tightly together to solve problems in a cooperative parallel fashion. Therefore, one of the criteria to judge the suitability of an interconnection network for the implementation of parallel computers is whether the network can be laid out compactly in a VLSI grid. Huge number of the processor are connected to each other there are no guaranteed the obtained solution is desirable. Here adding the concept of neural, adaptive linear elements proposed by Widrow added minimize the error.

2. BACKGROUND OF NEURAL NTEWORKS

The study of artificial neural networks has been inspired in part by the observation that biological learning system are built of very complex webs of interconnected neurons. Typically, the human brain consists of approximately 10 11 neurons, each with an average of 10 3 - 10 4 connections. It is believed that the immense computing power of the brain is the result of the parallel and distributed computing performed by these neurons.

The transmission of signals in biological neurons through synapses is a complicated chemical process in which specific transmitter substances are released from the sending side of the synapse. The effect is to raise or lower the electrical potential inside the body of the receiving cell. The neuron fires if the potential reaches a threshold. This is the characteristic that the artificial neuron model proposed by McCulloch and Pitts attempts to reproduce.

Back propagation is the standard training method that is applied to multi-layer feed-forward networks. The algorithm consists of passing the input signal forward and the error signal backward through the network. In the forward pass, one input vector is presented to the input layer, and the input signal is filtered forward through the subsequent layers of the network to generate a set of outputs. The network outputs are compared with the actual outputs of the training example and an error signal is generated. In the backward pass, the synaptic weights are updated using the learning constant according to the effect they have in generating the incorrect outputs [12], [13].

This neuron model is widely used in artificial neuron networks with some variations (Figure 1).

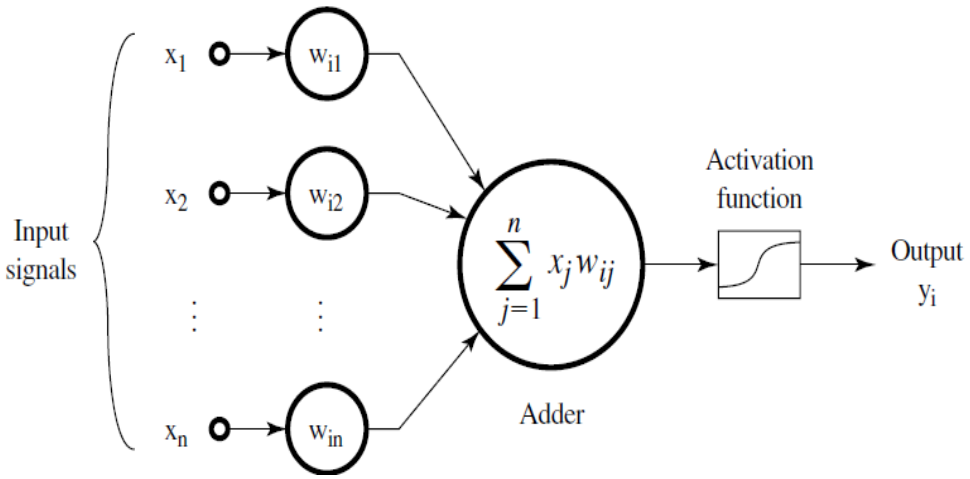


Figure: 1 Artificial neuron model

The artificial neuron presented in Figure 1 has N inputs, denoted as $x_1, x_2 \ldots x_n$. Each line connecting these inputs to the neuron is assigned a weight, denoted as $w_1, w_2 \ldots w_n$, respectively. The action, which determines whether the neuron is to be fired or not, is given by the formula:

$$a = \sum_{j=1}^{n} w_j x_{ij}$$

The output of the neuron is a function of its action:

$$Y_i = f(a)$$

Originally the neuron output function *f (a)* proposed in McCulloch-Pitts model was a threshold function. However, linear, ramp and sigmoid functions are also widely used today. An ANN system consists of a number of artificial neurons and a huge number of interconnections among them. According to the structure of the connections, two different classes of neural network architectures are identified.

In layered neural networks, the neurons are organized in the form of layers. The neurons in one layer get input from the previous layer and feed their output to the next layer. This type of network is called *feedforward neural network*. The first and last layers are *input layer* and *output layer* respectively, and the layers that are not input or output are called {\it hidden layers}. Networks with one or more hidden layers are called *multi-layer networks*. *Multi-Layer perceptron* is a well-known feedforward layered neural network, on which the Backpropagation learning algorithm is implemented [9], [10], [11].

3. NEURAL NETWORKS BASED A FRAMEWORK FOR EMBEDDED INTERCONNECTIO NETWORK

Proposed a Framework model-

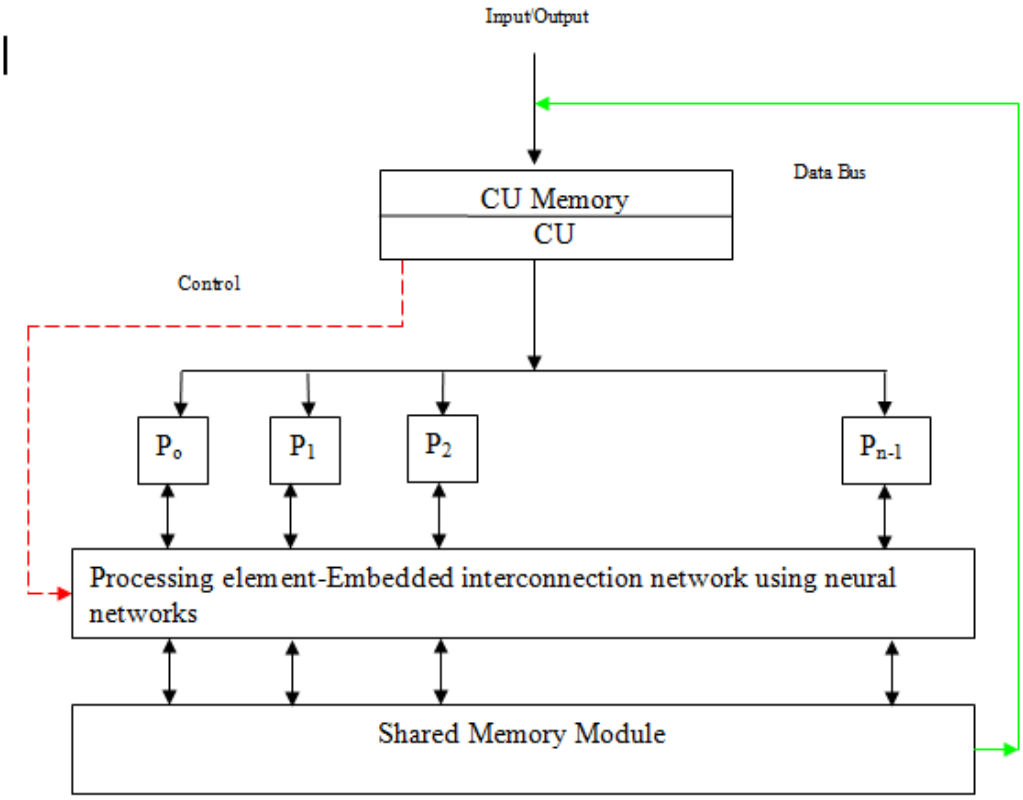


Figure: 2 A Framework of SIMD

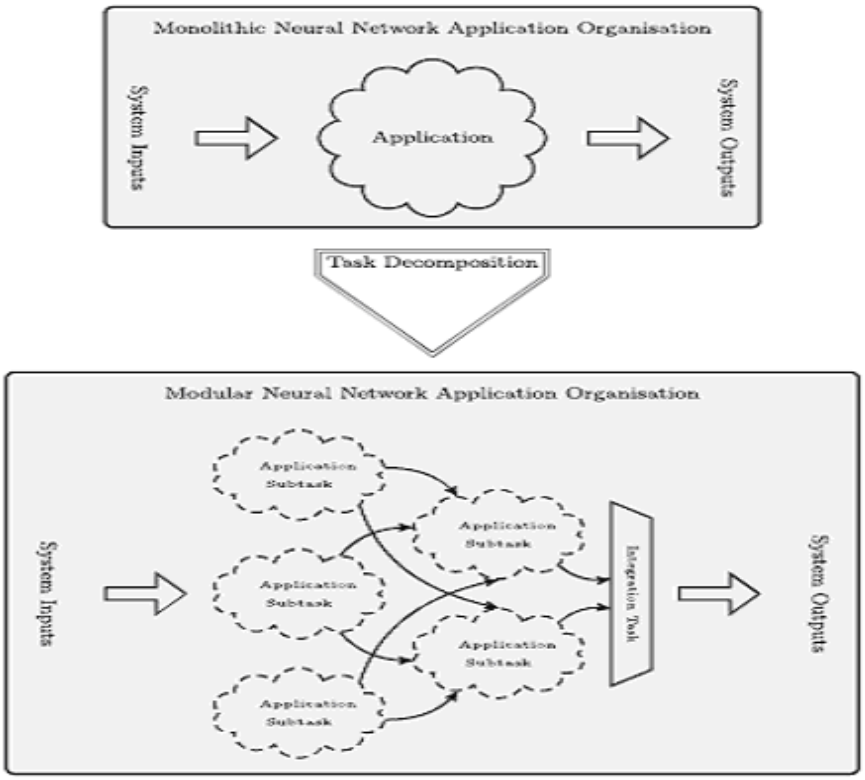


Figure: 2(a) Modular Network and Design Methodology

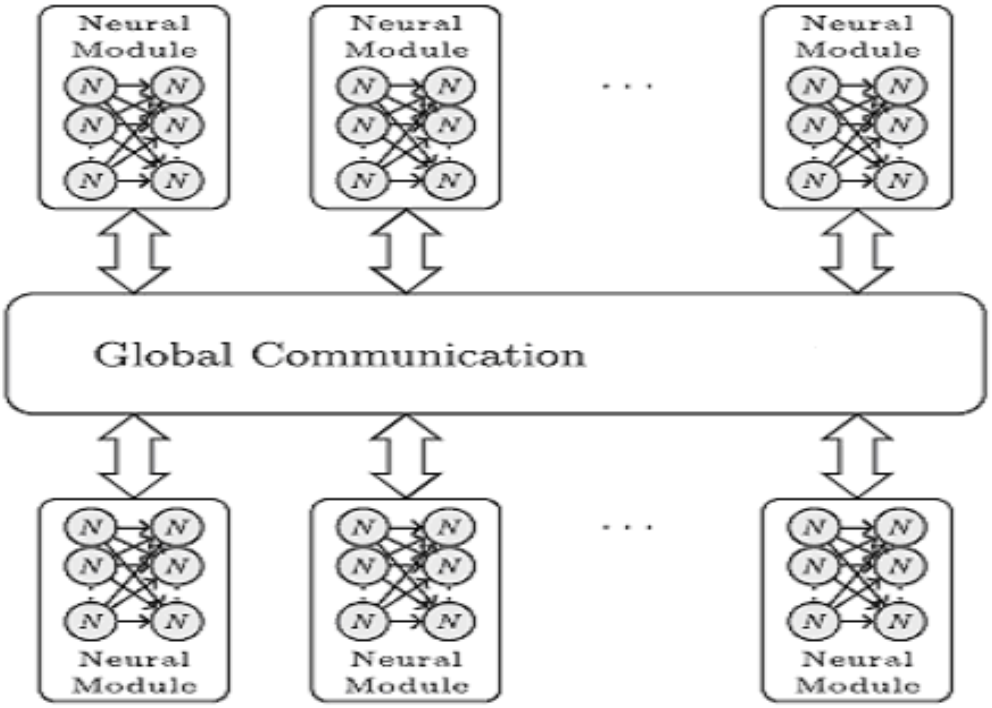


Figure: 2 (b) Modular Neural Network and Execution Architecture

Components:

1. Control unit
2. Processing element
3. Shared memory model
4. Emphasized on interconnection network
    - Static network
    - Embedded network
        - Architecture of neural network (Modular Network and Design Methodology , Modular Neural Network and Execution Architecture**)**
    - Prefer the interconnection on based neural (Adaline Algorithm)

In figure 2, Feedforword Network with embedded hypercube is used in as Interconnection network in parallel computing system. These artificial models rely heavily on highly interconnected computational units functioning in parallel Illustrate in figure 3 (c) A Feedforword 3-3-3-3 Network and in figure 3(b) and 3(a) are rooted tree structure with and without embedded hypercube.

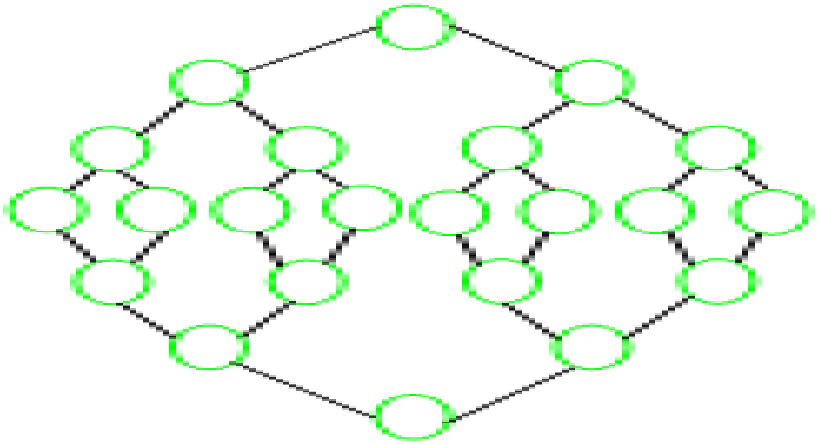


Figure: 3 (a) A Rooted tree Network (without embedded hypercube)

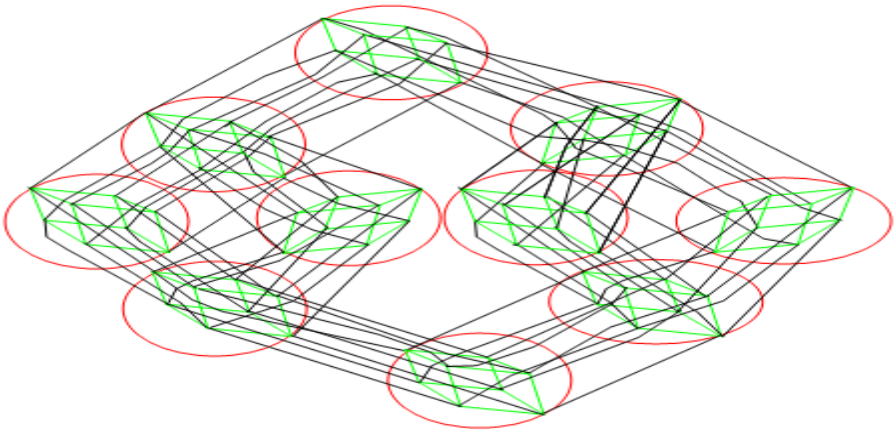


Figure: 3 (b) A Rooted tree Network (with embedded hypercube)

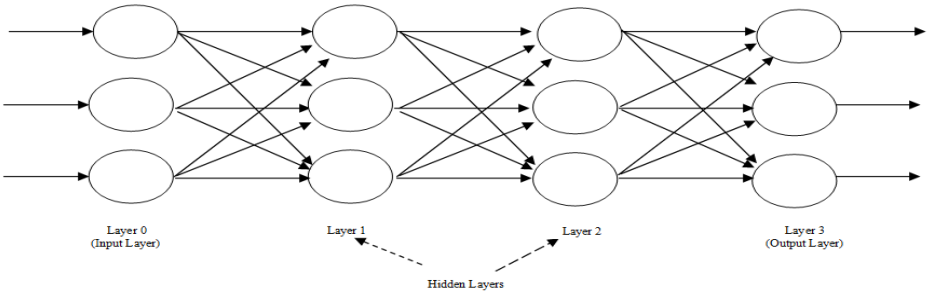


Figure: 3 (c) A Feedforword 3-3-3-3 Network (without embedded hypercube)

4. A FEEDFORWORD- NETWORK WITH EMBEDDED HYPERCUBE:

The artificial neuron is like that the processing element in parallel computing system presented in Figure 4 , including the red circle shown in the figure 4, node1, node2, node 3, and node 4 . There were the two case arises to provide the input at end and obtain the output from the other end in the interconnection network.

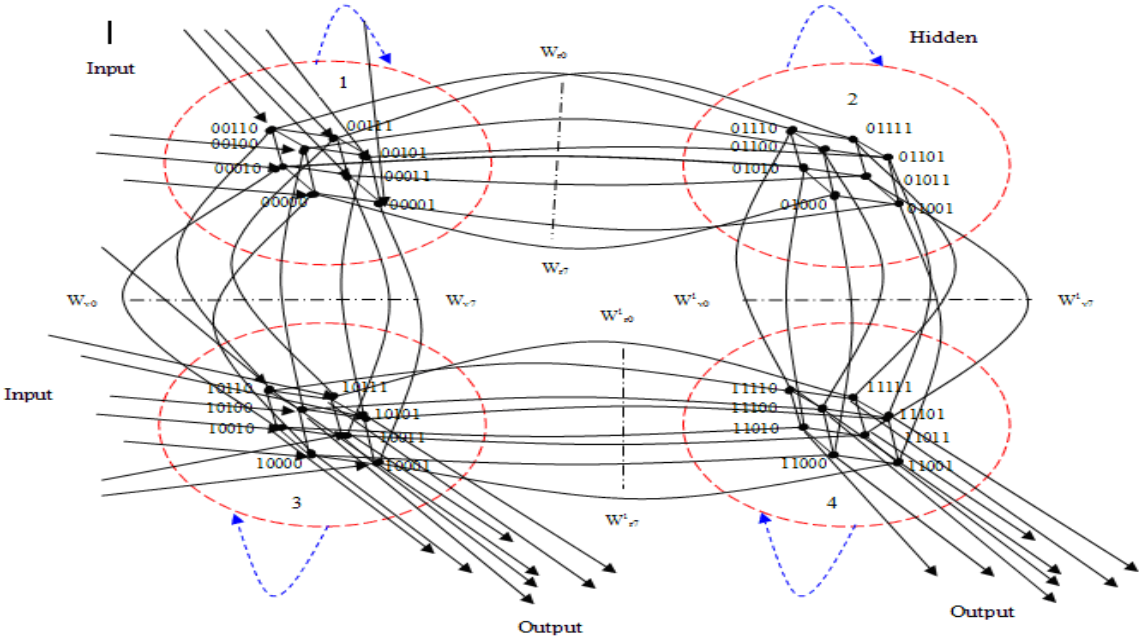


Figure: 4 A Feedforword- Network with embedded hypercube

Case: 1 At the Node 1 to apply the input. Node1 embedded hypercube (2^3) that means eight input to apply Node1 enclosed with red dotted circle. S is the finite set of containing input $N = \{n_1, n_2 \ldots n_8\}$. Enclosed dotted circles Node 2 are hidden node and Enclosed dotted circle Node 4 are output node with embedded hypercube.

Case: 2 At the Node 3 to apply the input. At node 3 embedded hypercube (2^3) that means eight input to apply Node 3 enclosed with red dotted circle. S is the finite set of containing input N= {$n_1, n_2 \ldots n_8$}. Enclosed dotted circle Node 3 are itself output node with embedded hypercube.

A Feedforword- Network with embedded hypercube has $\mathbf{n_1}$ inputs, denoted as x_{ij} (i.e. $x_{11}, x_{12}, \ldots, x_{1n}$) where i=1 and j = 1 to n in general form . Each line connecting these inputs to the neuron or processing element assigned a weight, denoted as w_{ij} (i.e. $w_{11}, w_{12} \ldots w_{1n}$) where i=1 and j = 1 respectively in general. The action, which determines whether the processing element is to be fired (active processing element) or not (inactive), is given by the formula:

$$a = \sum_{i=1, j=1}^{n} w_{ij} x_{ij}$$

The output of the processing element is a function of its action:

$$Y_i = f(a)$$

Defined $Y_I = 1$ where f (a) =1 processor are Active, $Y_I = 0$ where f (a) =0 processor are Inactive.

4.1. Routing Technique selection of Active and Inactive processing Element: A Neural Network Based Approach

Design a network structure is generally performed using an embedded methodology or a series of techniques. Many such techniques exist, all with the common objective to assist improve the performance of system required. Although some neural net architecture think that just choose one architecture apply the embedded technique is applicable to all situations, one methodology or technique cannot possibly be sufficient for all conditions.

Embedded select a particular static network technique for any combination of three reasons: (1) It is the only technique that the embedded knows, (2) It is the embedded favorable methodology for all situations, (3) The embedded understands intuitively that the technique is effective in the current circumstance. Clearly the third reason represents the better way. Here neural network is used for embedding in neural net embedding technique to select the active (ON) or passive (OFF) processing element in interconnection network in parallel computing. Proposed Model is implemented in MATLAB. The NNtool of MATLAB has been used to implement the proposed Model.

4.2. Proposed Algorithm: Neural network based model for Embedded Technique

a) Initially the techniques and their impact are analyzed from small scale embedded system and large scale embedded system.

b) Adjacency matrix A is obtained for small scale projects and , matrix B is obtained for large scale projects

c) Using MATLAB tool.

- Using nntool for creating neural network
- Assign input vector (v) values
- Assign target values (t)
- Create neural network
- Training the neural network
- The result of training
- Simulating the neural network

d) Analysis of result obtained.

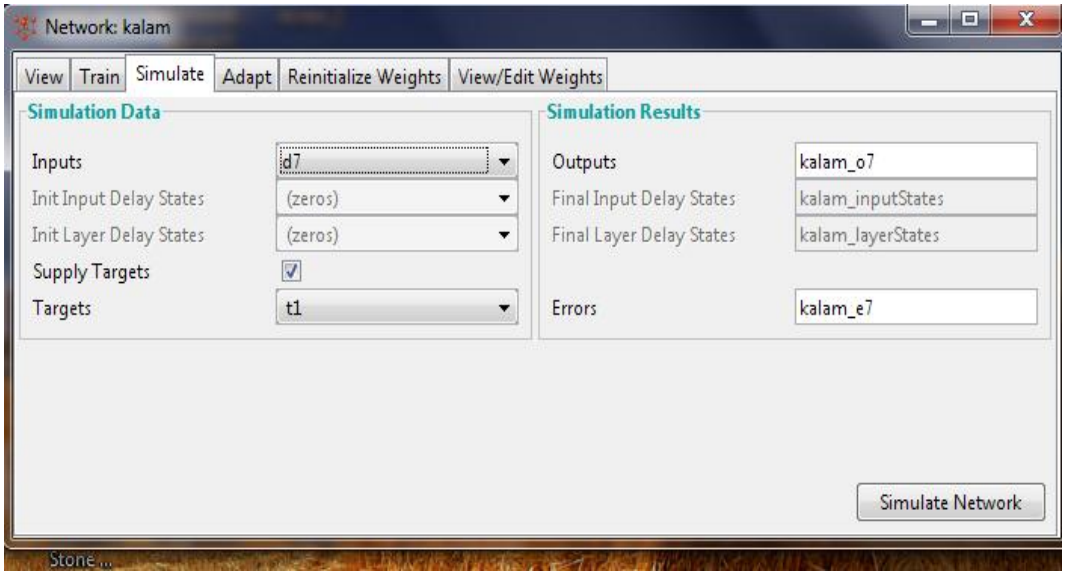


Figure: 5 (a) Simulation Process

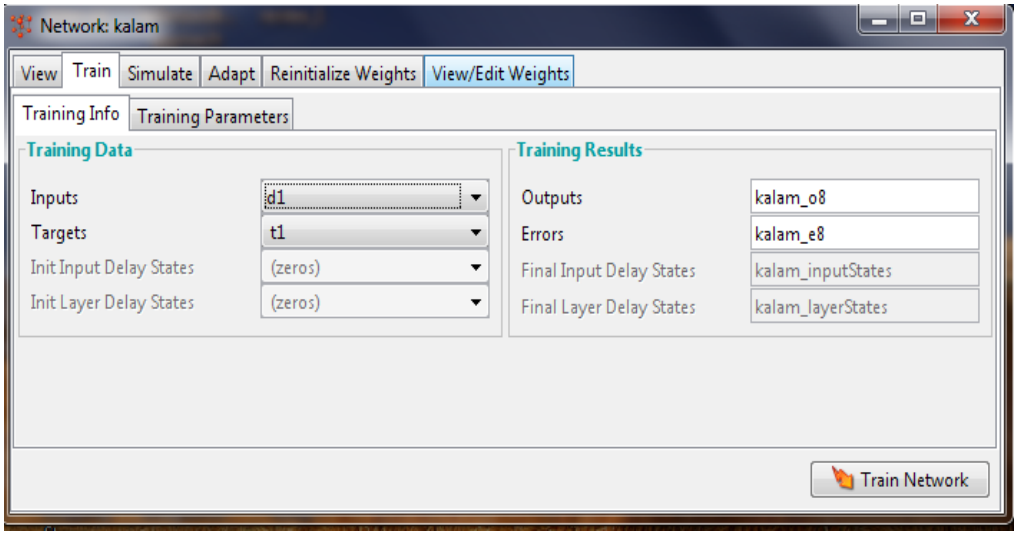


Figure: 5 (b) Training Process

4.3. Description of Algorithm:

Method followed for Routing Technique to select active or passive element.

1. Initially the Table1 and Table 2 of Routing techniques and their impact are analyzed for active and passive.

2. Matrix A is obtained from small scale cubic structure and Matrix B is obtained from large scale hypercube.

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Matrix A

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Matrix B

3) Using MATLAB for Masking Technique to select active or passive element.

a) Using nntool for creating neural network
b) Assign input vector (v) values
c) Assign target values (t)

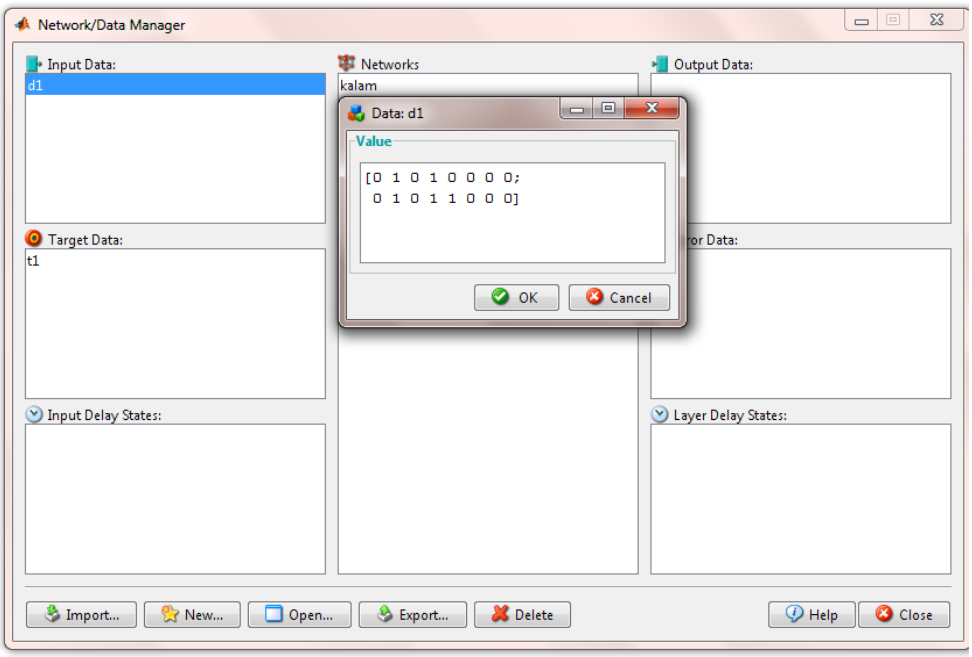


Figure6: Assign Input values to neural Network

c) Create neural network with following:-

- Name: Embedded Network
- Network properties

i. Network Type: Feed forward Back propagation

ii. Input Range: [0,1;0,1]

iii. Training Function: TRAINLM

iv. Adaptation Learning Function: LEARNGDM

v. Performance function: MSE

vi. No of Layers:

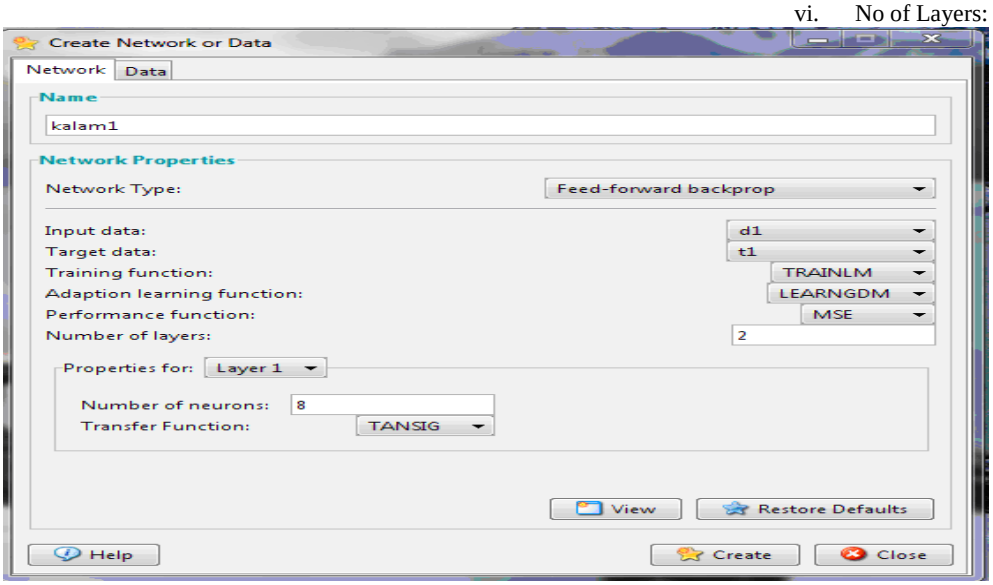


Figure: 5 Create neural network

e) Neural Network is shown in figure-

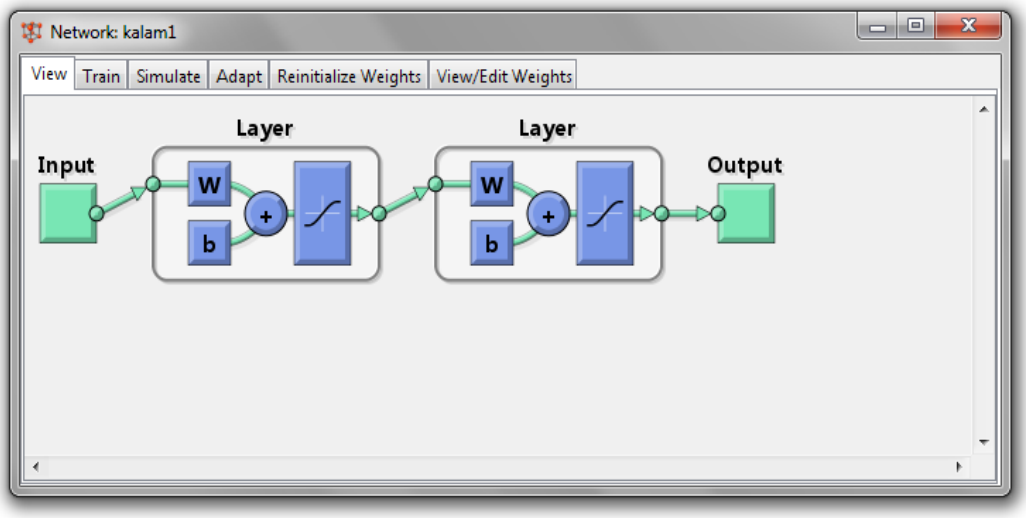


Figure: 6 View of neural network

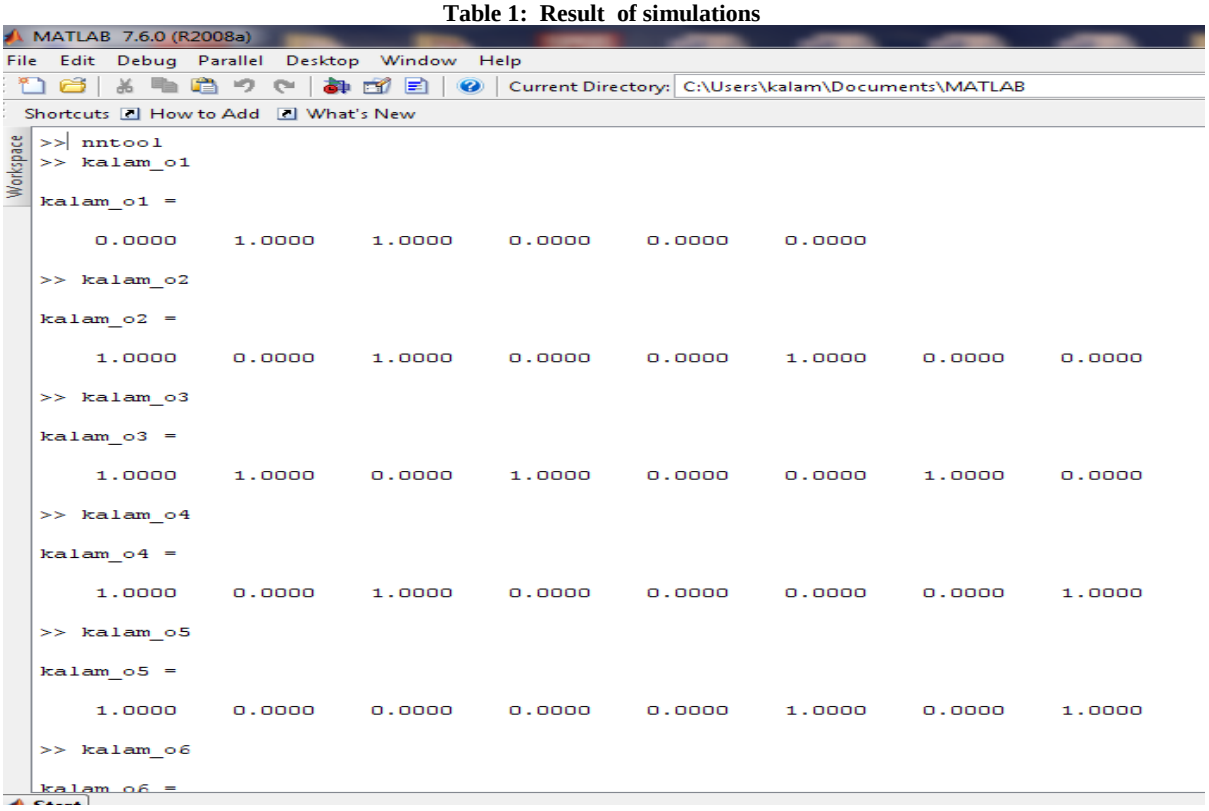
4.4. Analysis of Result

MATLAB is used for embedding hypercube on routing technique. The routing algorithm is implemented in nntool box. The nntool box is used for creating neural network. Assign input vector (v) values V={0 1 0 1 0 0 0 0 ;0 1 0 1 1 0 0 0 } and assign target values (t)

t= {0 1 0 1 1 0 0 0}.

The Network Type is Feed forward Back propagation with Input Range [0, 1; 0, 1] and TRAINLM as the Training Function. The Adaptation Learning Function is LEARNGDM and Performance function as MSE with No of Layers is 2.

Table 1: Result of simulations



```
kalam_o6 =

    0.0000    1.0000    0.0000    0.0000    1.0000    0.0000    1.0000    0.0000

>> kalam_o7

kalam_o7 =

    0.0000    0.0000    1.0000    0.0000    0.0000    1.0000    0.0000    1.0000

>> kalam_o8

kalam_o8 =

    0.0000    0.0000    0.0000    1.0000    1.0000    0.0000    1.0000    0.0000

>>
```

The neural network is trained, outcome of training is obtained, and further neural network is simulated, and performance indicates in table 1. Then the result is analyzed as '1' in the output vector will indicate the routing technique is used for the Active , '0' in the output vector will corresponds the technique indicated is Inactive for the given evaluating of routing factor R .

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

5. CONCLUSION

The traditional approach for quantifying neural network hardware performance is to measure the numbers of many accumulative operations performed in the unit time and the rate of Weight updates. These two measurements somewhat correspond to the MIPS measured on traditional systems. The implementations of the neural architecture are different sizes. To apply the routing function on neural net to routing of data by each neuron (processing element). To embedding the lower class of hypercube in static topology to provide the route by routing function and embedding the large class of hypercube in the same above structure to obtain the final routing factor for two incidence matrix formed by lower class and large class embedding hypercube in the same static architecture**.** The neural concept used in this paper to improve the performance of lower scale to higher scale of k-ary n cube architecture.

6. REFERENCES

[1] K.Hwang, "*Advanced Computer Architecture,"Parallelism, Scalability, Programmability",* New York McGraw-Hill, 1993.

[2] Kim Jong-Seok, Lee Hyeong-Ok and Heo Yeong-Nam**,** "Embedding among HCN(n,n), HFN(n,n) and hypercube" **,** Proceedings of Eighth International conference on parallel and distributed systems (ICPADS)-2001 pp 533 – 540, 26-29 June 2001.

[3] A. Louri and H. Sung, "An Optical Multi-Mesh Hypercube: A Scalable Optical Interconnection Network for Massively Parallel Computing," *IEEE J. Lightwave Technology*, vol. 12, pp. 704–716, Apr. 1994.

[4] N. Gopalakrishna Kini, M. Sathish Kumar and Mruthyunjaya H.S., "A Torus Embedded Hypercube Scalable Interconnection Network for Parallel Architecture," IEEE explore conference publications, Mar.2009, pp.858-861.

[5] Ahmed Louri and Hongki Sung, "An Optical Multi-Mesh Hypercube: A Scalable Optical interconnection Network for Massively Parallel Computing", Journal of Light wave Technology, Vol.12, No.4, Apr.1994, pp.704-716.

[6] N. Gopalakrishna Kini, M. Sathish Kumar and Mruthyunjaya H.S," Performance Metrics Analysis of Torus Embedded Hypercube Interconnection Network", International Journal on Computer Science and Engineering Vol.1 (2), 2009, 78-80.

[7] L.M. Ni, and P.K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Networks", IEEE Computer, vol.26, no.2, pp. 62-76, Feb. 1993.

[8] Kai Hwang, F.A.Briggs,"Computer Architecture and Parallel Processing ", McGraw-Hill International edition.

[9] Yihua Liao," *Neural Networks in Hardware: A Survey".*

[10] C. M. Bishop," *Neural Networks for Pattern Recognition",* Claren- don Press, Oxford, 1995.

[11] W. Maass and C. M. Bishop," *Pulsed Neural Networks.* Bradford Books/MIT Press", Cambridge, MA, 2001.

[12] W. Gerstner and W. Kistler," *Spiking Neuron Models: Single Neurons, Populations, Plasticity*", Cambridge University Press, Cambridge, 2002.

[13] J-Y. Mignolet, S. Vernalde, D. Verkest, and R. Lauwereins," En- abling Hardware Software Multitasking on a Recon_gurable Com- puting Platform for Networked Portable Multimedia Appliances. In *Proceeding," The 2002 International Conference on Engineering of Reconfigurable Systems and Algorithms*. June 2002.

[14] Mohd.Kalamuddin Ahmad, Mohd. Husain," Required Delay of Packet Transfer Model for Embedded Interconnection Network", International Journal of Engineering Research & Technology (IJERT) Vol. 2 Issue 1, January- 2013 ISSN: 2278-0181.

[15] J.F.Fang, J.Y.Hsiao and C.Y.Tang, "Embedding Meshes and TORUS Networks onto degree-four chordal rings," IEE Proc.-Comput. Digit. Tech., Vol.145, No.2, Mar.1998.

[16] Mohd.Kalamuddin Ahmad, Mohd. Husain, A.A. Zilli*," A Statistical Analysis and Comparative Study of Embedded Hypercube", International Journal of Computer Applications (0975 – 8887) Volume 103 – No 16, October 2014.*