IJCSN International Journal of Computer Science and Network, Volume 5, Issue 1, February 2016
ISSN (Online) : 2277-5420 www.IJCSN.org
Impact Factor: 1.02

Hopfield Neural Networks for Aircrafts' Enroute Sectoring: KRISHAN-HOPES

Dr. Krishan Kumar

Department of Computer Science, Faculty of Technology, Gurukul Kangri University
Haridwar-249404, Uttarakhand, India

Abstract - Air traffic controlling is a very complex task for the air port personnel. Hence the emphasis on some new advance computing techniques had always been a great and important area of research. Hopfield neural networks or simply Hopfield nets, a widely used popular category of feedback neural network or recurrent neural networks may play a very important role in handling issues related to air traffic control. As Hopfield nets provide a model for the memory of human brain and therefore they can memorize the input patterns of any real life problem. Hence these nets can be efficiently and effectively used for the air space sectoring problem. In this paper, a way to divide the existing space scenario in different sectors using Hopfield nets is presented. It is found that this method is appropriate for making the sectors of a congested busy air space. The result shows that algorithm gives the near optimal solution for 48 nodes or aircrafts.

Keywords - *Artificial Neural Network, Hopfield Network, Air Space, Sectoring, Collision Avoidance.*

1. Introduction

Air transportation is the best and most important mode of travel by passengers nowadays. It is a good way to save travelling time of passengers. Air traffic can be divided usually in two special categories i.e. ground movement traffic control and enroute traffic control. Ground movement control includes aircraft landing, takeoff, runway allocation, ground clearance, runway clearance etc. While enroute air traffic includes issues like collision avoidance, finding shortest route, sectrorization, communication with neighboring aircrafts etc. [1],[2].

Earlier the issues related to ground movement control like air traffic runway allocation had been handled using many soft computing techniques by Krishan Kumar et al. [3]. On the other hand enroute traffic issues like collision avoidance and sectoring are also of great concern, and must be handled carefully. Nowadays the problem of congestion in air space or enroute is a big problem for airport authorities as well as for air traffic controllers. Hence there is a great need to solve this problem. Many approaches to this kind of air space problem have been proposed [4].

Delahaye et al. successfully applied genetic algorithms for regrouping of sectors [5], [6], [7]. Genetic algorithms for conflict resolution had given good results. When joining two airports, an aircraft must follow routes and beacons. These beacons are necessary for pilots to know their position during navigation; and because of the small number of beacons on the ground they often represent crossing points of different airways. A try based on fuzzy logic and expert knowledge had been developed, for which daily operational data about decisions of experts were used. "If then" rules were found and the automation decision tool was developed [8].

Generally at the dawn of civil aviation, pilots resolve conflicts themselves, because they always fly aircrafts in good visibility conditions. On the other hand modern jet aircraft do not enable pilots to resolve conflicts because of their high speed and their ability to fly with bad visibility. Therefore, pilots must be helped by air traffic controller who has a global view of the current air traffic distribution in the airspace and can give orders to the pilots to avoid collisions. But when many aircrafts simultaneously present in the space, a single controller is not able to manage all the traffic. At most of the airports, airspace is partitioned into sectors, each of them being assigned to an air traffic controller. Sectoring is done in an empirical way by some airspace experts who apply the rules they have learned with their experience. After some time every sector when overloaded must be modified as soon as possible, usually due to traffic congestion. One can try to improve and complete the process of sectoring with an automatic approach in order to give a solution to the sectoring problem in the whole airspace and that can be later refined by experts [9].

This paper presents a popular feedback neural network's category Hopfield nets to manage the problem of sectoring [10]. It is a kind of neural network which can store memory patterns. Basically all the neurons make a network like mesh topology in which every neuron is connected to other neuron. All the neurons make loops except the self loops. Therefore these networks are also known as recurrent or dynamic networks. Furthermore Hopfield network has the ability to adjust not only themselves but also the neighboring neurons too. This is called cooperative learning. Earlier the same problem of aircraft sectoring has been solved successfully by Krishan Kumar using the two other very well known unsupervised learning techniques i.e. adaptive resonance theory map (ART1) neural networks [11] & self organizing map (SOM) neural network [12]. In other words we can say that these three techniques Hopfield nets, Kohenen's SOM, and Grossberg's ART1 are best suited for the correlation and classification. Ealier ART1 neural network has successfully been applied for aircraft's collision avoidance [13].

2. A Simplified Aircraft Enroute Model

2.1 Introduction

Since it takes much time to train an air traffic controller on his sector (3 to 4 months), it must not be investigated a real time sectoring optimization according to the variations of the traffic load. Instead a registered maximum load traffic period on the working network is considered. The problem is then to partition the air space into sub sectors to get a balanced control workload. When examining the physical air traffic network, it is noted that airways are superposition of several routes which have the same projection on the floor, but at different altitudes. According to their semi circular rule, an airway can be modeled by a bidirectional link which gathers several individual aircraft routes "Fig. 1".

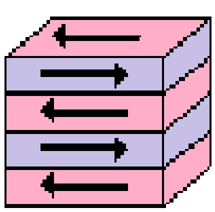

Fig. 1 Airways modeling [14]

An air traffic network is considered in a two dimensional space with flows on it inducing workload distributed over the space. This workload must be equilibrated convex sectors in a way that minimizes coordination. This sectoring takes some constraints into account [14], coming from the air traffic control system, which are given below:

A pilot must not encounter twice the same controller during his flight to prevent useless coordination; this means that an aircraft crossing a sector will encounter only two sector frontiers.

A sector frontier has to be at least at a given distance from each network node (safety constraint). As a matter of fact, when a controller has to solve a conflict, he needs a minimum of time to elaborate a solution. Each controller managing individually his sector, if a sector frontier is too close to a crossing point, he is not able to solve any conflict because he has not enough time between coordination step and the time the aircraft reaches the crossing point. The minimum delay time is fixed at 15 minutes, which can be converted into a distance knowing the aircraft speed.

An aircraft has to stay at least a given amount of time in each sector it crosses to give enough time to the controller to manage the flight in good conditions (min stay time constraint).

2.2 Mathematical Formulation

Transportation Network: Transportation network is defined as a doublet (N,L) in which N is the set of nodes (with their positions in a topological space) and L is the set of links each of them transporting a quantity $f_{i,j}$ of flow from node i to node j [15].

Construction of Sectors: According to the previous section, the sectors to build are to be convex (with a polygonal shape included with the convexity property). To reach this goal, Forgy Aggregation method, from dynamic clustering in exploratory statistics can be used, which aims at extracting clusters from a set of points randomly distributed in a topological space [15]. This method randomly throws K points (the class centers) in the space domain containing the transportation network and aggregates all domain points to their nearest class centers. This method ends up in a K partitioning of our domain into convex sectors with linear frontiers "Figure 2".

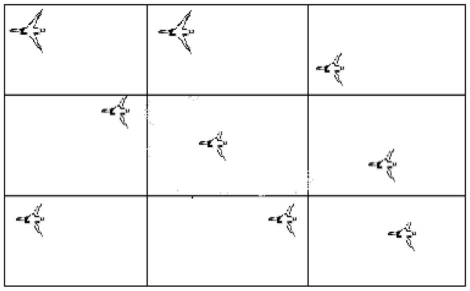

Fig. 2 Nine sectors without congestion

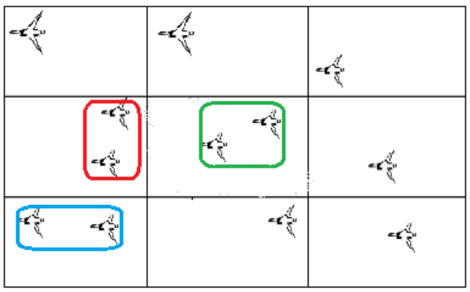

Fig. 3 Sectors 4,5,7 are in congestion

Workload In A Induced Sector: Here quantitative criteria are taken into account to compute controller's workload. According to the controllers themselves, workload can be divided into three parts which correspond respectively to the conflict workload, the coordination workload, and the trajectories monitoring workload of the different aircraft which are present in a sector. The conflict gathers with different actions of the controller to solve conflicts. The monitoring aims at checking the different trajectories of the aircraft present in the sector and induce the workload [16].

3. Problem Description

The problem which is to be solved can be divided in two different parts corresponding to different goals:

A: Equilibrium of the different sectors workload according to the number of aircrafts and conflicts in each sector.

B: Minimization of the coordination workload.

The second criterion is typically a discrete graph partitioning problem with topological constraints. Having chosen a continuous flow representation the first criterion, the discrete continuous problem is also NP_HARD [17]. Moreover this kind of problem may have many optimal solutions due to the different possible symmetries in the topological space. One must be able to find all of them because they have to be refined by experts and not known at this step which one is really the best. This last point makes rejection of classical simulated re-annealing optimization which updates only one state variable, even if it might give better results in some cases [18].

On the other hand, Hopfield network maintains and improve a numerous population of states variable according to its associative memory capability and would be able to find near optimal solutions. Therefore learning in using Hopfield network seems to be relevant for airspace sectoring problem.

4. Introduction to Hopfield Networks

Basically artificial neural networks are biologically inspired. They are inspired from human brain, which is the combination of millions of small processors distributed throughout the brain and performs the typical tasks after learning. Or we can say that artificial neural networks perform computational tasks by modeling the human brain activities. Generally the neural networks are divided in two categories, one is of feedforward network and other is feedback network. Flow of signals in feedforward is in forward direction only. Feedforward networks can be represented by directed graph. On the other hand in feedback networks the output is sent backward towards the input neuron which forms a cycle or loop. Such networks are unsupervised fully connected, symmetrically-weighted that extended the ideas of linear associative memories by adding cyclic connections. Neurons with either a hard-limiting activation function or with a continuous activation function can be used in such systems It is a form of recurrent artificial neural network popularized by John Hopfield, a physicist in 1982, but described earlier by Little in 1974 [10], [19].

The physicist Hopfield showed that models of physical systems could be used to solve computational problems. In the beginning of 80s Hopfield published two scientific papers, which attracted much interest. This was the starting point of the new era of neural networks, which still continues today [20], [21].

The core question: How is one to understand the incredible effectiveness of a brain in tasks such as recognizing a particular face in a complex scene? Like all computers, a brain is a dynamical system that carries out its computations by the change of its 'state' with time. Using these collective properties in processing information, simple models of the dynamics of neural circuits are described that have collective dynamical properties. These can be exploited in recognizing sensory patterns. It is effective in that it exploits the spontaneous properties of nerve cells and circuits to produce robust computation. Identifying words in natural speech is a difficult computational task which brains can easily do. They use this task as a test-bed for thinking about the computational abilities of neural networks and neuromorphic ideas.

The importance of the Hopfield nets in practical application is limited due to theoretical limitations of the structure, but, in some cases, they may form interesting models.

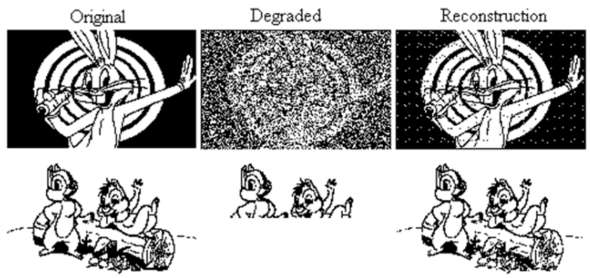

Fig. 4. Hopfield network reconstructing degraded images from noisy (top) or partial (bottom) cues [20].

4.1 Architecture

Conceptually Hopfield networks contain three important things or elements: basin of attraction, attractors, and trajectories. Furthermore Hopfield nets serve as content-addressable memory systems with binary threshold nodes in which an item can be accessed by knowing just part of it content (like Google or other search engines). It becomes very robust in case of hardware damage. They are guaranteed to converge to a local minimum, but convergence to a false pattern (wrong local minimum) rather than the stored pattern (expected local minimum) can occur. They also provide a model for understanding human memory. It does not contain self loops. These nets have two applications. First, they can act as associative memories. Second, they can be used to solve optimization problems. The behavior of such a dynamical system is fully determined by the synaptic weights.

The weights in the Hopfield network are constrained to be symmetric, i.e., the weight from neuron i to neuron j is equal to the weight from neuron j to neuron i. There are no self-connections, so $w_{ii} = 0$ for all i. It can also be thought as an Energy minimization process as shown in figure 5. For the figure 5, the weights can also be shown in form of matrix as given below:

$$\begin{pmatrix} 0 & w12 & w13 & w14 & w15 \\ w21 & 0 & w23 & w24 & w25 \\ w31 & w32 & 0 & w34 & w35 \\ w41 & w42 & w43 & 0 & w45 \\ w51 & w52 & w53 & w54 & 0 \end{pmatrix}$$

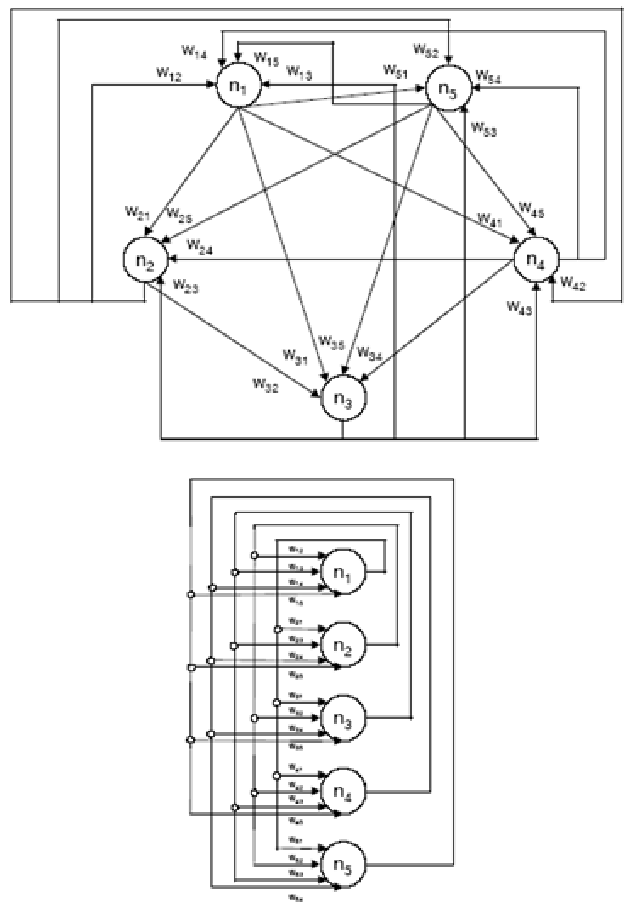

Fig. 5 Architecture of a Hopfield network

4.2 Algorithm

1: Assign connection weights

$$W_{ij} = \begin{cases} \frac{1}{N}\sum_{s=0}^{M-1} x_i^s x_j^s & i \neq j \\ 0 & i = j \end{cases} \quad (1)$$

Where W_{ij} is the connection weight between node *i* and node *j*, and x_i^s is the element i of exemplar patterns, and is either +1 or -1, in total.

2: Initialize with unknown pattern

$$\mu_i(o) = x_i \quad 0 \leq i \leq N-1 \quad (2)$$

Where $\mu_i(t)$ is the output of node *i* at time *t*.

3: Iterate until convergence is achieved:

$$\mu_i(t+1) = f_h\left[\sum_{i=0}^{N-1} w_{ij}\ \mu_j(t)\right] \quad 0 \le j \le N\text{-}1 \quad (3)$$

where the function f_h is the hard-limiting non-linearity, the step function.

Repeat the iteration until the outputs from the node remain unchanged.

Now given the weight matrix and the updating rule for neurons the dynamics of the network is defined if we tell in which order we update the neurons. There are two ways of updating them:

Asynchronous: one picks one neuron, calculates the weighted input sum and updates immediately. This can be done in a fixed order, or neurons can be picked at random, which is called asynchronous random updating.

Synchronous: the weighted input sums of all neurons are calculated without updating the neurons. Then all neurons are set to their new value, according to the value of their weighted input sum. The lecture slides contain an explicit example of synchronous updating.

4.3 Storing Memories

If we use activities of 1 and -1, we can store a state vector by incrementing the weight between any two units by the product of their activities. It is to be noted here that we must treat biases as weight from a permanently on unit.

$$\Delta w_{ij} = s_i s_j \qquad (4)$$

With states of 0 and 1 the rule is slightly more complicated.

$$\Delta w_{ij} = 4\,(s_i - \tfrac{1}{2})(s_j - \tfrac{1}{2}) \qquad (5)$$

4.4 Spurious Minima

Each time we memorize a configuration, we hope to create a new energy minimum. But what if two nearby minima merge to create a minimum at an intermediate location? Basically this limits the capacity of a Hopfield net. Using Hopfield's storage rule the capacity of a totally connected net with N units is only 0.15N memories. This does not make efficient use of the bits required to store the weights in the network.

4.5 Another Computational Nets of Hopfield Nets

Instead of using the net to store memories, use it to construct interpretations of sensory input. The input is represented by the visible units. The interpretation is represented by the states of the hidden units. The badness of the interpretation is represented by the energy.

4.6 Code Used in C language for Initialization

```
//---GENERATING THE HOPFIELD NETWORK -----
void GenerateNetwork(NET* Net)
{
 int i;
 Net->Units  =  N;
 Net->Output  =  (int*)  calloc(N, sizeof(int));
 Net->Threshold  =  (int *)  calloc(N, sizeof(int ));
 Net->Weight  =  (int **) calloc(N, sizeof(int *));

 for (i=0; i<N; i++)
 {
  Net->Threshold[i] = 0;
  Net->Weight[i]  = (int *) calloc(N, sizeof(int));
  }
 }

void CalculateWeights(NET* Net)
{
 int i,j,n;
 int Weight;

 for (i=0; i<Net->Units; i++)
 {
  for (j=0; j<Net->Units; j++)
  {
   Weight = 0;
    if (i!=j){
              for (n=0; n<NUM_DATA; n++)
              {
              Weight += Input[n][i] * Input[n][j];
              }
             }
    Net->Weight[i][j] = Weight;
   }
  }
}

//----SETTING THE INPUT FOR DELAY STATES---
void SetInput(NET* Net, int * Input)
{
 int i;

 for (i=0; i<Net->Units; i++)
```

```
 {
  Net->Output[i] = Input[i];
  }
 WriteNet(Net);
 }

//---------- CALCULATING THE OUTPUT ----------
void GetOutput(NET* Net, int * Output)
{
 int i;

 for (i=0; i<Net->Units; i++)
 {
  Output[i] = Net->Output[i];
 }
 WriteNet(Net);}
```

5. Hopfield Network for Problem Optimization

As Hopfield nets mechanism minimize an energy function, therefore we can also map various interesting problems on Hopfield nets. If Hopfield nets are used for function optimization, the objective function F to be minimized is written as energy function (F) in the form of computational energy (E). The comparison between E and F leads to the design, i.e. definition of links and biases, of the neural net that can solve the problem. Biological data processing problems often involve an element of constraint satisfaction in scene interpretation, for example, one might wish to infer the spatial location, orientation, brightness and texture of each visible element, and which visible elements are connected together in objects. These inferences are constrained by the given data and by prior knowledge about continuity of objects.

6. Result & Discussion

A classic constraint satisfaction problem to which Hopfield networks have been applied is the travelling salesman problem. Which states that set of cities is given, and a matrix of some distances between those cities is given. The task is to find a closed tour of the cities, visiting each city once that has the smallest total distance. The travelling salesman problem is equivalent an NP-complete problem.

Furthermore, to investigate the performance of the Hopfield algorithm, 48 aircrafts in a 2D plane are used. The algorithm is applied to divide the congested space containing 48 aircrafts flying in air space. Earlier the positions of these aircrafts were congested, which can be seen in figure 3. To overcome this problem we have used 48 neurons in all. All 48 input vectors can be shown by using the six binary bits. Because the maximum input limit is 48, which is greater than 32 and less than 64, therefore the maximum input value is 64. “Table 1” shows the equivalent binary pattern used for the first 24 inputs.

The neural network of 48 neurons is trained in MATLAB till the stable states achieved. For each epoch all the 48 patterns are submitted to the designed Hopfield net. For this we have made an assumption that we follow shortest route among nine sectors then aircrafts are not in congestion. This is similar to the travelling salesman problem to find a shortest route visiting every node/city once.

Table 1: Binary Vectors for Input Pattern

S.No.	*Input #*	*Equivalent Binary Vector*	*Equivalent Bipolar Vector*
1	I_1	[0 0 0 0 0 1]	[-1 -1 -1 -1 -1 1]
2	I_2	[0 0 0 0 1 0]	[-1 -1 -1 -1 1 -1]
3	I_3	[0 0 0 0 1 1]	[-1 -1 -1 -1 1 1]
4	I_4	[0 0 0 1 0 0]	[-1 -1 -1 1 -1 -1]
5	I_5	[0 0 0 1 0 1]	[-1 -1 -1 1 -1 1]
6	I_6	[0 0 0 1 1 0]	[-1 -1 -1 1 1 -1]
7	I_7	[0 0 0 1 1 1]	[-1 -1 -1 1 1 1]
8	I_8	[0 0 1 0 0 0]	[-1 -1 1 -1 -1 -1]
9	I_9	[0 0 1 0 0 1]	[-1 -1 1 -1 -1 1]
10	I_{10}	[0 0 1 0 1 0]	[-1 -1 1 1 -1 -1]
11	I_{11}	[0 0 1 0 1 1]	[-1 -1 1 -1 1 1]
12	I_{12}	[0 0 1 1 0 0]	[-1 -1 1 1 -1 -1]
13	I_{13}	[0 0 1 1 0 1]	[-1 -1 1 1 -1 1]
14	I_{14}	[0 0 1 1 0 0]	[-1 -1 1 1 -1 -1]
15	I_{15}	[0 0 1 1 1 1]	[-1 -1 1 1 1 1]
16	I_{16}	[0 1 0 0 0 0]	[-1 1 -1 -1 -1 -1]
17	I_{17}	[0 1 0 0 0 1]	[-1 1 -1 -1-1 1]
18	I_{18}	[0 1 0 0 1 0]	[-1 1 -1 -1 1-1]
19	I_{19}	[0 1 0 0 1 1]	[-1 1 -1 -1 1 1]
20	I_{20}	[0 1 0 1 0 0]	[-1 1 -1 1 -1 -1]
21	I_{21}	[0 1 0 1 0 1]	[-1 1 -1 1 -1 1]
22	I_{22}	[0 1 0 1 0 1]	[-1 1 -1 1 1 -1]
23	I_{23}	[0 1 0 1 1 1]	[-1 1 -1 1 1 1]
24	I_{24}	[0 1 1 0 0 0]	[-1 1 1 -1 -1 -1]

Assume that we want to build an interpolative memory with three input neurons and three output neurons. We have the following three exemplars (desired input-output pairs):

Step-1: convert [0 0 0 0 0 1] to [-1 -1 -1 -1 -1 1] (This input is converted into a matrix and find its transpose)

Step-2: Find the multiplication of both the matrices

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

Step-3: The resultant matrix is the weight matrix

$$\left\{ \left(\begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix} \right), \left(\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \right), \left(\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 5 \\ 2 \\ 8 \end{bmatrix} \right) \right\}$$

Then

$$W = \begin{bmatrix} 3 \\ 3 \\ 3 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 5 \\ 2 \\ 8 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

$$W = \begin{bmatrix} 0 & -3 & 0 \\ 0 & -3 & 0 \\ 0 & -3 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 3 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 5 \\ 0 & 0 & 2 \\ 0 & 0 & 8 \end{bmatrix} = \begin{bmatrix} 1 & -3 & 5 \\ 2 & -3 & 2 \\ 3 & -3 & 8 \end{bmatrix}$$

If we set the weights w_{mn} to these values, the network will realize the desired function. So if we want to implement a linear function $R^N \rightarrow R^M$ and can provide exemplars with orthonormal input vectors, then an interpolative associative memory is the best solution. It does not require any training procedure, realizes perfect matching of the exemplars, and performs plausible interpolation for new input vectors. Of course, this interpolation is linear. Usually, the vectors x_p are not orthonormal, so it is not guaranteed that whenever we input some pattern x_p, the output will be y_p, but it will be a pattern similar to y_p. Since the Hopfield network is recurrent, its behavior depends on its previous state and in the general case is difficult to predict.

After completion of training, same patterns are used for simulation. All the input patterns and their respective stable states, which are actually nine sectors, have been implemented. For each epoch, the total strength of the weight for each neuron is calculated. After 95 epochs network attains its optimal values i.e. stable states based on the memory recalling. The neuron with the asymptotic stable state which is the local minimum for the energy dissipation function, finally declared as winner. Initial weight matrix is given by the figure 6.

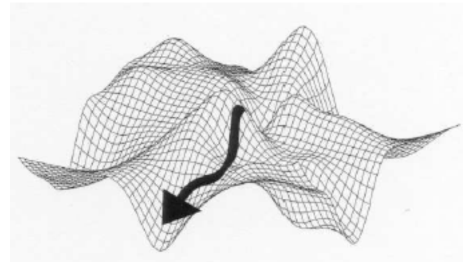

Fig. 6 Initial weight positions

Nine sectors in airspace are used, each sector is considered as a different state or class, and aircrafts in the particular sector are presented as an input pattern.

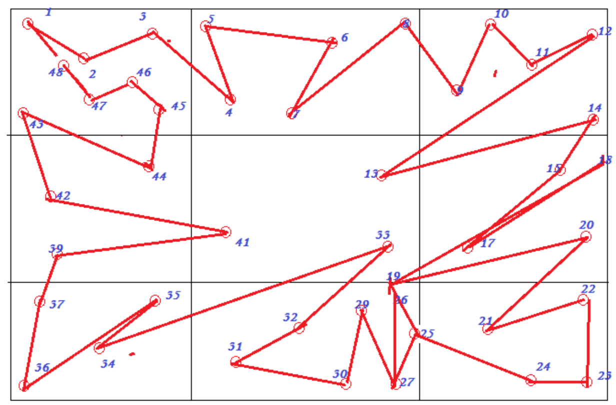

Fig. 7 Optimized path

7. Conclusion

Consequently I have simulated the results for the 48 aircrafts flying in the air at same height and same horizontal plane. In “Table 1” only 24 binary patterns are shown. Next 24 patterns i.e from 25 to 48 shall be converted on the same pattern. Earlier they were in congestion due to crossing the limit of maximum possible aircrafts in a sector. Hopfield neural net has been successfully applied and implemented using MATLAB; and It seems suitable for solving the aircrafts’ space congestion problem. The problem of enroute congestion is similar to the most popular operation research travelling salesman problem (TSP). Hence the division of aircrafts into different sectors on the basis of their positions can also be done. This really improved the algorithm performances regarding the position of aircrafts from their earlier positions. Also the results can be easily obtained using the Hopfield nets. This approach is also one of the ways to find an optimal or near optimal solution in contrast with the other unsupervised learning algorithms i.e. adaptive resonance theory and self organizing map unsupervised neural networks. In future, on the basis of these results by Hopfield nets, other problems like to find the shortest path in wireless sensor networks, mobile adhoc networks, and vehicular networks can also be easily implemented.

References

[1] Bowers, Peter M., Boeing Aircraft since 1916, ISBN 0-85177-804-6, London: Putnam Aeronautical Books, 1989.

[2] E.P Gilbo, "Optimizing airport capacity utilization in air traffic flow management subject to constraints at arrival and departure fixes", IEEE Transactions on Control Systems Technology, Vol. 5, no. 5, sep. 1997.

[3] K. Kumar, R. Singh, Z. Khan, and A. Indian, "Air Traffic Runway Allocation Problem Using ARTMAP (ART1)", Ubiquitous Computing and Communication International Journal, Korea, Vol. 3, No 3, July, 2008, pp. 130-136.

[4] Min Xue, "Airspace Sector Redesign Based on Voronoi Diagrams", University of California at Santa Cruz, Moffett Field, CA 94035, 2008.

[5] Daniel Delahaye, Jean-Marc Alliot, Marc Schenauer, and Jean-Loup Farges, "Genetic algorithms for partitioning air space", Proceedings of the 10th Conference on Artificial Intelligence and Application, IEEE, 1-4 Mar 1994, pp. 291 – 297.

[6] D. Delahaye, M. Schoenauer and J. M. Alliot, "Airspace sectoring by evolutionary computation", Evolutionary Computation Proceeding by IEEE World Congress on Computational Intelligence, IEEE International Conference on Vol., Issue, 4-9 May 1998.

[7] B. Pesic and D. Delahaye, Daily Operational airspace sector grouping. April 27, 1999.

[8] O babic and T Kristic, "Airspace daily operational Sectorization by fuzzy logic", Elsevier, 2000.

[9] Riley, V. Chatterji, G. Johnson, W. Mogford, R. Kopardekar, P. Sieira, E. Landing, and M. Lawton, G. "Pilot Perceptions of Airspace Complexity", Part-2, Digital Avionics Systems Conference. DASC 04, IEEE, 2004.

[10] Laurene Fausett, Fundamentals of Neural networks: Architecture, Algorithm and Applications, Pearson Education, ISBN 978-81-317-0053, 1994.

[11] Krishan Kumar, "ART1 Neural Networks for Air Space Sectoring", International Journal of Computer Applications, USA, 2012, pp. 20 – 24.

[12] Krishan Kumar, "Self Organizing Map (SOM) Neural Networks For Air Space Sectoring", in IEEE International Conference Computational Intelligence & Communication Networks, Udaipur, India, 17-18, Nov'2014.

[13] K. Kumar, R. Singh, Z. Khan, "Air Traffic Enroute Conflict Detection Using Adaptive Resonance Theory Map Neural Networks (ART1)", Ubiquitous Computing and Communication International Journal, Korea, Vol. 3, No 3, July, 2008, pp. 28-35.

[14] D. Klingman and J. M. Mulvey editors, Network models and associated applications, North Holland, ISBN: 0-444-86203-X, 1981.

[15] Gilbert Saporta, Probabilistic analysis of statistic techniques, 1990.

[16] P.L. Tuan, H.S. Procter, and G.J. Couluris, "Advanced productivity analysis methods for air traffic control operations", FAA report RD-76-164, Standford Research Institute, Mento Park CA 94025. December 1976.

[17] Chung-Kuan Cheng, "The optimal partitioning of networks", Networks, 22:297-315, 1992.

[18] Lester Ingber and Bruce Rosen, "Genetic Algorithm and fast simulated re-annealing: a comparison", Mathematical and Computer Modeling, 16(1):87-100, 1992.

[19] Kevin, An Introduction to Neural Networks: Routledge. ISBN 1857285034, 2002.

[20] "Neural networks and physical systems with emergent collective computational abilities", Proceedings of the National Academy of Sciences, 1982, pp. 2554-2558.

[21] "Neurons with graded response have collective computational properties like those of two-state neurons", Proceedings of the National Academy of Sciences, 1984, pp. 81:3088-3092.

Dr Krishan Kumar is Assistant Professor in the Department of Computer Science, Faculty of Technology, Gurukula Kangri University, Haridwar, Uttarakhand, India. His area of interest is artificial neural networks (ANN) and its applications. He obtained his M.C.A degree from IMS Ghaziabad, India and Ph. D.(CS & IT) from Institute of Engineering & Technology, M.J.P.Rohilkhand University, Bareilly. He has also qualified National Eligibility Test (UGC-NET) in Computer Science and Application. Dr. Kumar has authored more than 30 research papers in international/national Journals/ Conferences. Presently he is vice chairman of Computer Society of India, Haridwar Chapter and elected Chairman for the session 2016-17. He has also been working as reviewer for many reputed journals like Elsevier, Springer etc. for last 10 years.