International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

AN APPROACH TO DESIGN AND IMPLEMENTATION OF DYNAMIC GEOMETRIC TRAVELLING SALESPERSON PROBLEM USING HOPEFIELD NEURAL NETWORK

Yogeesha C. B, Dr. Ramachandra. V. Pujeri
Karpagam University, Coimbatore,
Tamilnadu, INDIA – 641 021
Collaboration Technology Group (CTG),
yogeesha_cb@yahoo.com ; ycb@cisco.com sriramu_vp@kggroup.com

ABSTRACT

The use of Artificial Neural Network was originally motivated by the astonishing success of these concepts in their biological counterparts. Despite their totally different approaches, it can merely be seen as optimization methods, which is used in a wide range of applications, where traditional methods often prove to be unsatisfactory. Dynamic Geometric Travelling Salesperson Problem (DGTSP), which considered being a classic example for Combinatorial Optimization problem. Given the positions of a shortest tour that starts and finishes at the same city in which the cities are dynamically varied. The DGTSP is simple to state but hard to solve exactly, in that there is no known method of finding the optimum tour, shortest of computing the length of every possible tour and then selecting the shortest one. It is said to be NP-Complete. This paper explores an approach to design and implement this problem using Continuous Hopfield Neural Network Model.

Index Terms—Combinatorial Optimization Problem, Artificial Intelligence, Artificial Neural Networks, TSP, Geometric Dynamic TSP, Neurodynamic Hopefield Neural Network Model.

1. INTRODUCTION

In Neurodynamic Hopefield Network Model [3][4], the most general case, neural networks consist of a (often very high) number of neurons, each of which has a number of inputs, which are mapped via a relatively simple function to its output. Networks differ in the way their neurons are interconnected (topology), in the way the output of a neuron determined out of its inputs (propagation function) and in their temporal behavior (synchronous, asynchronous or continuous). While the temporal behavior is normally determined by the simulation hard and software used and the topology remains very often unchanged, the propagation function is associated with a set of variable parameters which refer to the relative importance of the different inputs (weights) or to describe a threshold-value of the output (bias).

The most striking difference between neural networks and traditional programming is, that neural networks are in fact not programmed at all, but are able to ”learn" by example, thus extracting and generalizing features of a presented training set and correlating them to the desired output. After a certain training period, the net should be able to produce the right output also for new input values, which are nor part of the training set. This learning process is accomplished by a training algorithm, which successively changes the parameters (i.e. the weights and the bias) of each neuron until the desired result, typically expressed by the maximum distance between the actual and the training output is achieved. Those algorithms can’t be subdivided into two major groups according to the data used to update the neuron parameters. Local algorithms (e.g. Perceptron Learn Algorithm (PLA), Backpropagation) restrict themselves to the local data at the in and outputs of each neuron, while global methods (e.g. Simulated Annealing) also use overall data (e.g. statistical information).

The Travelling Salesperson Problem (TSP) is a classic optimization problem that defines easy solution. The TSP problem is NP-complete problem. The conventional approach of comparing the cost function for alternate solutions and picking the most optimum, fails in the case of TSP because of the enormously large number of alternate solutions that need to be examined. Thus any algorithm for this

International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

problem is going to be impractical with certain examples. However the artificial neural network provides a possible technique to solve the TSP.

The Travelling Salesman Problem [3] is a classified *NP-hard* optimization problem. It can be defined as:

Input:

- Set C co-ordinates of *n* cities {v1,...,vn}.
- Distances d(i,j) for each pair (vi,vj) of C.

Output:

-Hamiltonian cycle for C.

2. NEURAL HOPEFIELD NETWORKS AND GEOMETRIC-TSP

Neural networks [4][8] have been used to solve a variety of constrained optimization problems. Due to their massive parallelism and fast network convergence to optimal solutions, they reduce the time that usually arise in most sequential algorithms. Solving optimization problems requires minimization of some cost functions subject to a set of constraints. These cost functions are known as energy functions, and the neural network will produce good solutions by minimizing an energy function.

This paper is motivated by basis of "Solution for the Geometric Travelling Salesperson Problem Using Continuous Hopfield Network Model" [3][6]. This is proposed as an artificial neural network hopefield network for solving difficult optimization problems like the Travelling Salesman Problem (TSP). In a Hopfield neural network each processing element has an external input and has weighted connections from other neurons. The continuous Hopfield model is used to find solutions for the 10-city problem. The 10 co-ordinates used as an input to the problem. Like CityA(x1,y1), CityB(x2,y2)...CityJ(x10,y10). The distances between the cities are calculated using discrete Euclidean length:

$$Dist_{x,y} = Co\_ordinatres_{x1,y1,x2,y2} = \sqrt{(x1 - x2)^2 + (y1 -$$

The distances between the cities are given nXn symmetric distance matrix. Where 'n' is the total number of cities. A salesperson has to visit 'n' cities in such a sequence, that the total distance traveled is a minimum. The travel is subject to the following constraints:

- The path is a Hamiltonian loop (the salesperson will return to the starting point)
- He shall visit each city once and only once.
- He cannot simultaneously be present in more than one city at any given time.
- The path of travel from one city to another is a straight line.

Consider the options available for 'n' cities in the list: The tour can begin from any city. Therefore we have 'n' possible starting points for the journey (however, the starting point is not a real issue. What matters is the sequence in which the cities are covered). After choosing the starting point, there are (n-1) options to choose the next city. After choosing the second city, there are (n-2) options to choose the third city. And so on... Thus, on the whole, n! distinct paths are possible. However, all the n! paths are not distinct (optimization algorithm need to be applied only for distinct paths).

- Since the optimization is aimed at the length of Hamilton loop, the starting point of the journey is not important. This reduces search arena (n-1)! possible paths.
- For every possible solution, its reverse sequence also gives the same distance (The path length is same for the sequence A→B→C→D→E→A, and also its reverse sequence A→E→D→C→B→A. This is true for every option).

Thus the number of paths that need to be examined are given by

$$N = \frac{n!}{2n} = \frac{(n-1)!}{2}$$

The problem is trivial if the number of cities is either 3 or less. However, for larger values of 'n' the number of paths that needs to be evaluated increases exponentially. For a 10 city tour, N=181,440 and for 11 cities N=1,814,400, and for 12 cities, N=19,958,400, for 30 cities, N=4.4x1030 : and for 100 cities, it is 93 326 215 443 944 152 681 699 238 856 266 700 490 715 968 264 381 621 468 592 963 895 217 599 993 229 915 608 941 463 976 156 518 286 253 697 920 827 223 758 251 185 210 916 864 000 000 000 000 000 000 000 000.

The basic procedure for solving the geometric TSP using a continuous Hopfield model [5] is described as follows:

Step 0: Initialize activation u[x][i] of all units and corresponding v[x][i].
Initialize network parameters A, B, C, D, N, α, Δt.
Set all unew[x][i] and vnew[x][i] to zero.

Step1: While the stopping condition is false, do Step 2 ~ Step 6.

Step 2: for x=1 to 10
for i=1 to 10

International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

Perform Step3 ~ Step4.

Step 3: Change activity on UNITx,i

unew[x][i] = u[x][i] + Δt * [-u[x][i] - A * Σv[x][j] <j!=i> - B * Σv[y][i] <y!=x> + C * {N-Σ Σv[.][.]} - D * Σ distance[x][y] * (v[y][i+1]+v[y][i-1]) <y!=x>].

Step 4: Apply output function vnew[x][i] = g(unew[x][i]).

Step 5: Update all u[x][i] and v[x][i] to unew[x][i] and vnew[x][i].

Set all unew[x][i] and vnew[x][i] to zero.

Step 6: Check stopping condition

The output function used is:

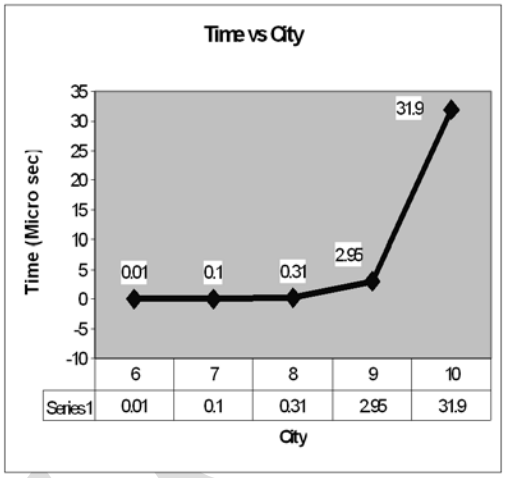
$$G(U_{x,i}) = \text{Output} V_{x,i} = 0.5 . (1 + \tanh(\alpha * UT[x,i])$$

The stop condition used is: The network was frozen. We define this as: The network was frozen if no outputs v[x][i] changed by more than 10E-10. One can also use the changes of u[x][i] as judgment. The algorithm was implemented using 'C' programming language under Linux platform. Figure 1, Figure 4 and Figure 5 are describes the results and comparison results of TSP.

	C1	C2	C3	C4	C5
C1	0	0	1	0	0
C2	1	0	0	0	0
C3	0	0	0	0	1
C4	0	1	0	0	0
C5	0	0	0	1	0

Cities (N)	Permutation Combination (N!) Iterations	Time (Micro Sec)
6	720	0.010000
7	5040	0.100000
8	40320	0.310000
9	362880	2.950000
10	3628800	31.90000

➲(I,j) (k,l)⬇	1	2	3	4	5	
1			$V_{1,3}$			A
2	$V_{2,1}$					B
3					$V_{3,5}$	C
4		$V_{4,2}$				D
5				$V_{5,4}$		E



Best tour for 10 Cities	Epochs (Length)	Time (Micro Sec)
Best Tour 1	127(2.947796)	0.020000
Best Tour 2	373(2.986585)	0.070000
Best Tour 3	158 (3.005290)	0.040000
Best Tour 4	134 (3.271382)	0.030000

3. DESIGN OF DYNAMIC GEOMETRIC TRAVELLING SALESPERSON PROBLEM (DGTSP)

The idea of Dynamic Traveling Salesperson Problem is obtained from "Traveling sales person problem". The report "Genetic Algorithms for Optimizing Neural Network Learning rate and Momentum with Emphasis on Geometric TSP" [3] and paper "Solution for the Geometric Traveling Salesperson Problem Using Continuous Hopfield Network Model" [6] are the main motivation for "Dynamic Geometric Salesperson problem". It is an enhancement of GTSP. Here main idea is to vary the number of cities during the tour of the sales person. TSP is a classic example for optimization problem that defines easy solution. The conventional approach of comparing the cost function for alternate solutions and picking the most optimum fails in the case of DTSP because of the enormously large number of alternate solutions that need to be examined. Thus any algorithm for this problem is going to be impractical with certain examples. However the Hopefield Neural Network Model provides a possible technique to solve DGTSP.

International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

The DGTSP problem is defined as: there is a list of cities initially that are to be visited by salesperson. These cities are varied during the tour, where actually the cities are dynamically varied at the time of tour. A salesperson starts from a city and does not visit the cities that are removed and visits the cities that bare added during the tour and he come backs to the same city after visiting all the cities. Here the objective is to find the path, which follows following constrains:

- The total length of the loop should be a minimum.
- The salesperson cannot be at two different places at the same time.
- The salesperson should visit each city only once.
- The salesperson should visit each city once and only once.
- The cities of the tour are dynamically varied.

In DGTSP, there are given n cities, and a non-negative distance between any two cities i and j. We try to vary cities during the travel of the salesperson and we find the tour for the salesperson that best fits the above mentioned criterion. There are various algorithms that can be used to try to solve such constrain problems. Most solution have used the following method [1] to find solution to TSP same can be used to solve DTSP:

- Hopfield Network Model
- Genetic Algorithms
- Kohonen Self-organizing map

For any N city problem, the distances between cities are calculated using discrete Euclidean length:

$$Dist_{x,y} = Co\_ordinatres_{x1,y1,x2,y2} = \sqrt{(x1-x2)^2 + (y1-}$$

The distances between the cities are given nXn symmetric distance matrix. Where 'n' is the total number of cities.

Hopefield Neural Network Model can be used for constrained optimization problems. They have several potential advantages over traditional techniques for certain types of optimization problems: they can find near optimal solutions quickly for large problems, they can also handle situations in which some constraints are weak (desirable, but not absolutely required).

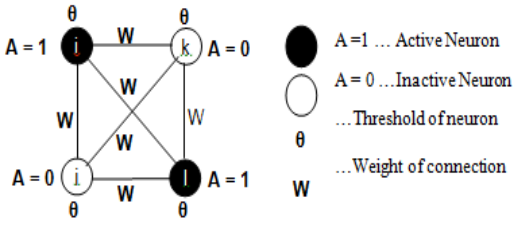


Figure 1 Simple example of Hopfield Network

A Hopfield network consists of binary neurons, which are connected by a symmetric network structure. Binary means that the neurons can be active ("ON", state 1) or inactive ("OFF", 0). The connections are weighted, and depending on the sign of the weight they can be intercepting or activating; e.g. a ON neuron activates all neurons, which are connected to it with a positive weight. There is a threshold value for every neuron, which the sum of the input values must reach to produce activity.

At the beginning of the calculation of the network output, the neuron's activation corresponds to the pattern to recognize. Then the network is iterated, which means that the state of the neurons is recalculated until the network is stable, i.e. the network state doesn't change any more. This is possible in a finite amount of time and iterations for Hopfield networks. This can also be seen as the minimization of the energy in the network, so that the final state is a minimum.

The basic procedure for solving the Dynamic TSP using a continuous Hopfield model[5] is described as follows:

Step 0: Initialize activation u[x][i] of all units and corresponding v[x][i].
Initialize network parameters A, B, C, D, N, α, Δt.
Set all unew[x][i] and vnew[x][i] to zero.
Step 1:While the stopping condition is false, do Step2 ~ Step6.
Step 2: for x=1 to N
Check for the increment or decrement of node and update u{x}{I] and v[x][I].
for i=1 to N
Perform Step3 ~ Step4.
Step 3: Change activity on UNITx,i
unew[x][i] = u[x][i] +
Δt * [-u[x][i] - A * Σv[x][j] <j!=i>
- B * Σv[y][i] <y!=x>
+ C * {N-Σ Σv[.][.]}
- D * Σ distance[x][y] *
(v[y][i+1]+v[y][i-1]) <y!=x>
].
Step 4: Apply output function vnew[x][i] = g(unew[x][i]).

International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

Step 5: Update all u[x][i] and v[x][i] to unew[x][i] and
vnew[x][i].
Set all unew[x][i] and vnew[x][i] to zero.
Step 6: Check stopping condition
The output function used is:

$$G(U_{x,i}) = \text{Output}V_{x,i} = 0.5 . (1 + \tanh(\alpha * UT[x,i]$$

The conventional Hopfield network is a single layer feed forward network without no self-feedback. For the specific application of n-city ('n' may vary) DGTSP, the single layer contains (nXn) neurons arranged in the form of an array. Here a whole row of neurons represents a city, and for the final solution, only one neuron in a row comes ON. Thus the live neurons in the successive columns indicate the sequence of visit (position of the city in the Hamiltonian loop). The Figure 2 shows a typical solution for a 5-city problem; here the optimum path is indicated as C2→C4→C1→C5→C3→C2.

For the given co-ordinates calculate the distance matrix by using

$$\text{Dist}_{x,y} = \text{Co\_ordinatres}_{x1,y1,x2,y2} = \sqrt{(x1 - x2)^2 + (y1}$$

Initialize Activation Matrix U and Output Matrix V
Activation Matrix

$$V_{i,j} = \sum_i \sum_j \frac{\text{Cities}}{100} + \text{RandomNoise}$$

$$\text{RandomNoise} = \frac{\left(\frac{\text{RAND}}{\text{RANDMAX}}\right)}{10.0} - 0.5$$

where random noise is between –0.05 to +0.05
Output Matrix

$$U_{i,j} = \sum_i \sum_j \text{atanh}\left(2 . \frac{V_{i,j\_1}}{\alpha}\right)$$

V = Activation Matrix
i = a specific row
j = a specific column

$$E_1 = \sum_x \sum_i \sum_{(j \neq i)} V_{x,i}$$

V = Activation Matrix
x = a specific row (xth city)
i = ith neuron in xth row
j = any neuron other than i in xth row

$$E_2 = \sum_x \sum_i \sum_{(y \neq x)} V_{y,i}$$

V = Activation Matrix
x = a specific column
i = ith neuron in xth column
y = any neuron other than i in the xth column

$$E_3 = \sum_x \sum_i \sum_j \sum_k V_{j,k}$$

V = Activation Matrix
x = a specific row
i = a specific column
j = any neuron in xth row
k = any neuron in ith column

$$E_4 = \sum_x \sum_i \sum_{(y \neq x)} \text{Dist}_{x,y} . (V_{y,i\_1} + V_{y,i+1})$$

Dist = Distance Matrix
V = Activation Matrix
x = a specific row
y = any row other than the xth row
i = any neuron in yth row (left to y or right to y)

$$\Delta U = \Delta T . (-1.0 . U_{x,i} - A . E_1 - B . E_2 + C . (N - E_3) - D . E_4)$$

UT and VT are temporary Activation Matrix and Output Matrix

$$UT_{x,i} = U_{x,i} + \Delta U$$

$$G(U_{x,i}) = \text{Output}V_{x,i} = 0.5 . (1 + \tanh(\alpha * UT[x,i])$$

$$VT_{x,i} = G(U_{x,i})$$

$$\Delta V = V_{x,i} - VT_{x,i}$$

Values of A, B, C, D, N, alpha (α) and deltat (Δt) are used as network parameters.

The initial activity levels (u[x][i]) were chosen so that Σ Σv[x][i] =10 (which is the desired total activation for a valid tour). To do this, at first, we assign each v[x][i] = 10/100 + some random noise = 0.1 + some random noise. The random noises are between [-0.05, +0.05].
Then we got u[x][i] by

$$U_{x,i} = \frac{\text{atanh}(2 . V_{x,i} - 1)}{\alpha}$$

Since the noises we added are random values, we can run the program many times with different starting configurations.

4. EXPERIMENTAL RESULTS

With the same network parameters and algorithm, but with different initial starting configurations of u[x][i] and v[x][i], we might get different training result, some might froze, some might fail to converge, and some might get valid tours. In our implementation, we try many different trails, none of

International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

them got frozen, some might fail to converge, and most of them can find a valid tour.

The result shows in figure 8 is the simulation using 10 cities, 11 cities and 12 cities. The traveling paths (Hamilton Loops) generated are shown in figure 8 with total length (distance travelled), total epoch and total time taken by the algorithm. Experiment is carried out from 2 cities to 12 cities.

Hopfield neural network is efficient and it can converge to stable states in hundreds times iterations. The output first gives the N-cities Co-ordinates from the user. i.e. the number of cities and their distance is calculated using Euclidean Length formula. Activation and output matrix are initialized first and calculated matrices of the same are displayed. Depend on the output and distance matrixes the tour route, total distance travelled is calculated. Which yields the DTSP solution is optimal.

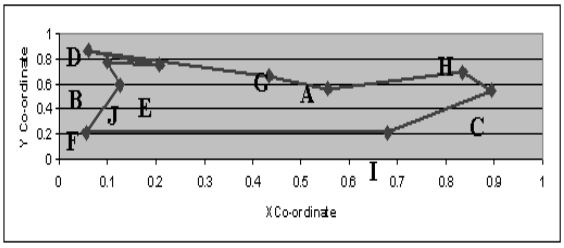


Figure 2 Location of 10 cities

Trail	Cities	Best Tour for Cities	Length	Epoch	Time (Micro Sec)
2	12	ABEGKFJLDICHA	4.604446	453	1.9899
4	11	HEBDAKJFICGH	4.205311	348	1.5182
7	10	DGAHCIFJBED	3.11461	110	0.2693

5. CONCLUSIONS AND FURTHER WORK

In Dynamic Geometric TSP approach, 94 % of test cases the algorithm converged, while in 4 % algorithm failed to converge and in remaining 2% the energy of the system increased instead of decreasing. As shown in Figure 8 the best optimal length obtained and time taken for 10 cities which converges is 110 epochs, with time 0.2693. And for 12 cities optimal length obtained is 453 epoch with 1.9899 time. With this results we can state that the time taken falls under the range n^2 to n!. In Nonrandomized city problem results shows in Figure 3 and Figure 4 if the number of cities increased, the permutation combinations are also increased by n!, so the Nonrandomized city problem falls in O (n!). The DGSTP approach in Figure 8 the time taken for best tour result in different runs, which is much less that of 7 cities run of Nonrandomized city problem in Figure 4. So result of GTSP [6] and DGTSP algorithm techniques to find shortest path is the best technique, where the time complexity falls in between O (n^2) and O (n!).

There is lot of scope for research work in the field of Artificial Intelligence, Genetic Algorithms, Neural Networks and Combinatorial Optimization. As the paper includes Artificial Intelligence further research can be carried on solving the same problem using White Blood Cells (WBC) [11]. WBC platelets contribute to protection against bacterial infection. The new research shows that when there aren't a lot of targets, cells do a pretty good job of finding the shortest possible route that hits all the targets. These cells "search" by tuning into local concentrations of chemical signals and following the signals to the nearest target. Repeating that process allows immune cells to find and demolish numerous attackers.

ACKNOWLEDGMENT

The author wish to thank member of Collaboration Test Group (CTG), Cisco Systems, Bangalore for their help in completing this work.

REFERENCES

[1] Elaine Rich and Kevin Knight, 1991, "Artificial Intelligence", Second Edition, Tata McGraw-Hill Publishing Company Limited, pp. 40-62.

[2] Byung-In Kim, Jae-Ik Shim, Min Zhang, December 1998 "Comparison of TSP Algorithms" Project for Models in Facilities Planning and Materials Handling

[3] Yogeesha CB and Vinaya Kumar K, 2004, "Genetic Algorithms for Optimizing Neural Network Learning Rate and Momentum pair with Emphasis on Geometric Traveling Salesperson Problem. " 2004, M.Tech. Project Report, NITK., Surathkal.

[4] Dan W. Patterson, 1996, "Artificial Neural Networks: Theory and Applications", Prentice Hall of India private Limited, pp. 14-19, 141-178

[5] The NIST website. National Institute of Standard and Technology, "Dictionary of Algorithms and Data Structures." [Online]. Available: http://www.nist.gov/dads/

[6] Yogeesha CB and Vinaya Kumar K, 2004 "Solution for the Geometric Traveling

International Journal of Technology and Engineering System (IJTES)
Vol 6. No.1 – Jan-March 2014 Pp. 58-64
©gopalax Journals, Singapore
available at : www.ijcns.com
ISSN: 0976-1345

Salesperson Problem Using Continuous Hopfield Network Model" 12th IEEE International Conference on Advanced Computing and Communication ADCOM 2004

[7] Dr. S N Omkar, Scientific Officer, Dept. of Aerospace Engg., IISc, Bangalore, 27th Aug. 2001 to 31st Aug. 2001, "Introduction to Artificial Neural Networks and Engineering Applications", Short term course jointly organized by Dept. of Aerospace Engg., IISc. Bangalore and Dept. of Computer Science, SIT., Tumkur. at SIT., Tumkur.

[8] Dan W. Patterson, 1996, "Artificial Neural Networks: Theory and Applications", Prentice Hall of India private Limited, pp. 14-19, 141-178

[9] Kate Smith, Marimuthu Palaniswami and Mohan Krishnamoorthy, NOVEMBER 1998 "Neural Techniques for Combinatorial Optimization with Applications", IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 9, NO. 6, pp. 1301-1318.

[10] Yogeesha CB and Dr. Ramachandra V Pujeri, 2012 "Randomized Algorithms: On the Improvement of Searching Techniques Using Probabilistic Data Structure Linear Linked Skip Lists", http://link.springer.com/chapter/10.1007/978-81-322-0740-5_19, International Conference on Advances in Computing, pp 147-153

[11] Rachel Ehrenberg, "White Blood Cells Solve Traveling-Salesman Problem": Online: http://www.wired.com/wiredscience/2011/04/cells-math-problem.