Two Pages Graph Layout Via Recurrent Multivalued Neural Networks

Domingo López-Rodríguez[1], Enrique Mérida-Casermeiro[1],
Juan M. Ortíz-de-Lazcano-Lobato[2], and Gloria Galán-Marín[3]

[1] Department of Applied Mathematics, University of Málaga, Málaga, Spain
{dlopez,merida}@ctima.uma.es
[2] Department of Computer Science and Artificial Intelligence, University of Málaga, Málaga, Spain
jmortiz@lcc.uma.es
[3] Department of Electronics and Electromechanical Engineering, University of Extremadura, Badajoz, Spain
gloriagm@unex.es

Abstract. In this work, we propose the use of two neural models performing jointly in order to minimize the same energy function. This model is focused on obtaining good solutions for the two pages book crossing problem, although some others problems can be efficiently solved by the same model. The neural technique applied to this problem allows to reduce the energy function by changing outputs from both networks –outputs of first network representing location of nodes in the nodes line, while the outputs of the second one meaning the half-plane where the edges are drawn.

Detailed description of the model is presented, and the technique to minimize an energy function is fully described. It has proved to be a very competitive and efficient algorithm, in terms of quality of solutions and computational time, when compared to the state-of-the-art methods. Some simulation results are presented in this paper, to show the comparative efficiency of the methods.

1 Introduction

In the last few years, several graph representation problems have been studied in the literature. Most of them are related to the linear graph layout problem, in which the vertices of a graph are placed along a horizontal "node line", or "spine" (dividing the plane into two half-planes or "pages") and then edges are added to this representation as specified by the adjacency matrix. The objective of this problem is to minimize the number of crossings produced by such a layout.

Some examples of problems associated to this linear graph layout problem (or 2 pages crossing number problem) are the bandwidth problem [1], the book thickness problem [2], the pagenumber problem [3,4], the boundary VLSI layout problem [5] and the single-row routing problem [6], or printed circuit board layout [7] and automated graph drawing [8]. In the case of designing a printed circuit board, for the case of

F. Sandoval et al. (Eds.): IWANN 2007, LNCS 4507, pp. 194–202, 2007.

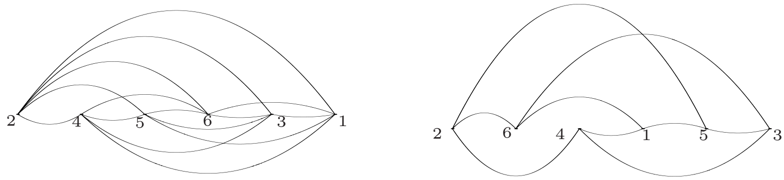

Fig. 1. Optimal linear layouts for K_6 and $K_{3,3}$

non-insulated wires, overlapping wires between electrical components may cause short circuits and thus may be avoided as much as possible.

Several authors study a restricted version of this problem in which the vertex order is predetermined and fixed along the node line, and edges are drawn as arcs in one of the two pages [9]. Other authors are more interested in the variant in which the node order is not fixed [10]. For this variant, it has been considered necessary to first find an optimal ordering of the vertices in order to compute the layout.

This problem is NP-hard [11,12]. So, many researchers have focused on finding efficient algorithms (some of them specially designed for the case of certain families of graphs) to solve the graph layout problem.

A comparison of several heuristics for this problem is presented in [9], including greedy, maximal planar, one-page, bisection and a neural heuristic, among others. Concretely, the neural model developed in [13] (and based on Takefuji and Lee's work [14,15]) was tested and obtained very good results, although authors indicate the possibility of non-convergence of this method. Due to the use of binary neurons, the model needs $2M$ neurons to represent the solution for a graph of M edges.

In this work we present a neural model designed to solve this problem. One of the differences of our model with the algorithms developed in literature is that there is no need of assigning a good ordering of the vertices at a preprocessing step. This optimal node order is computed by the model, as well as the relative position of the arcs.

Our model is a variant of the multivalued MREM model which has obtained very good results when applied to other combinatorial optimization problems [16,17,18,19], guaranteeing the convergence to local minima of the energy function.

2 Formal Description of the Problem

Let $G = (V, E)$ be an undirected graph where $V = \{v_i\}$ is the set of vertices and $E = (e_{i,j})$ is a symmetric binary matrix where $e_{i,j} = 1$ if the edge (v_i, v_j) exists.

The Two-page book Crossing Number Problem (2PLCN) consists in placing the graph nodes on a horizontal "node line" in the plane. Every edge can be drawn as an arc in one of the two half-planes (pages). The objective is to minimize the number of edge crossings. This problem belongs to the class of NP-hard optimization problems, even if nodes are fixed and only arcs can be drawn on the upper or lower half-plane.

An example of linear embedding of the complete graphs K_6 and $K_{3,3}$ with 3 and 1 crossings, respectively, is drawn in Fig. 1.

Fig. 2. Crossing condition $i < k < j < l$

2.1 Crossings Detection

Let us consider 4 positions in the node line verifying $1 \leq i < k < j < l \leq N$, where i, j, k and l are assigned to nodes v_i, v_j, v_k and v_l. Then, edges (v_i, v_j) and (v_k, v_l) are crossed if, and only if, both are represented (drawn) in the same half-plane.

In Fig. 2, we can observe that edges (v_i, v_j) and (v_k, v_l), represented in the node line and with endpoints $i < k < j < l$, produce a crossing, whereas if $i < j < k < l$ they do not, when both are represented in the same half-plane.

It seems reasonable to define $V_{v_a,v_b} = +1$ to indicate that the edge (v_a, v_b) will be represented in the upper half-plane, whereas $V_{v_a,v_b} = -1$, indicates that the arc will be drawn in the lower one. If the edge does not exist, we define $V_{v_a,v_b} = 0$.

These definitions allow us to define the number of crossings by means of the cost function:

$$C = \sum_{i} \sum_{k>i} \sum_{j>k} \sum_{l>j} \delta(V_{v_i,v_j}, V_{v_k,v_l})(1 - \delta(V_{v_i,v_j}, 0)) \tag{1}$$

where $\delta(x, y) = 1$ if $x = y$ and equals 0, otherwise (Krönecker delta function).

In Eq. (1), the term $\delta(V_{v_i,v_j}, V_{v_k,v_l})$ expresses that edges (v_i, v_j) and (v_k, v_l) will be drawn in the same half-plane, whereas $(1 - \delta(V_{v_i,v_j}, 0))$ indicates that the edge exists.

3 The Neural Model MREM

It consists in a series of multivalued neurons, where the state of i-th neuron is characterized by its output (v_i) that can take any value in any finite set $\mathcal{M}$. This set can be a non numerical one, but, in this paper, the neuron outputs only take value in $\mathcal{M} \subset \mathbb{Z}$.

The state vector $\boldsymbol{V} = (v_1, v_2, \ldots, v_N) \in \mathcal{M}^N$ describes the network state at any time, where N is the number of neurons in the net. Associated with any state vector, there is an energy function $E : \mathcal{M}^N \to \mathbb{R}$, defined by the expression:

$$E(\boldsymbol{V}) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} w_{i,j} f(v_i, v_j) + \sum_{i=1}^{N} \theta_i(v_i) \tag{2}$$

where $W = (w_{i,j})$ is a matrix, $f : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is usually a similarity function since it measures the similarity between the outputs of neurons i and j, and $\theta_i : \mathcal{M} \to \mathbb{R}$ is a threshold function. At each step, the state vector will be evolving to decrease the value of the energy function.

The cost function (number of crossings in the graph given by Eq. (1)), must be identified with the energy function of Eq. (2). As a result, we obtain $w_{i,j} = 1$ if $i < j$ and 0 otherwise. The similarity function $f(v_i, v_j)$ and the threshold θ_i can be expressed as:

$$f(v_i, v_j) = -2 \sum_k \sum_{l>k} \delta(V_{v_i,v_k}, V_{v_j,v_l})(1 - \delta(V_{v_i,v_k}, 0))$$

$$\theta_i(v_i) = -\sum_j w_{i,j} \sum_{k \leq j} \sum_{l>k} \delta(V_{v_i,v_k}, V_{v_j,v_l})(1 - \delta(V_{v_i,v_k}, 0))$$

To solve the 2PLCN problem we have considered two MREM neural models:

– The first network will be formed by N neurons, being N the number of nodes in the graph. Neurons output (the state vector) indicate the node ordering in the line. Thus, $v_i = k$ will be interpreted as the k-th node being placed in the i-th position in the node line. Hence, the output of each neuron can take value in the set $\mathcal{M}_1 = \{1, 2, \ldots N\}$.
– The second network will be formed by as many neurons as edges in the graph, M. The output of each neuron will belong to the set $\mathcal{M}_2 = \{-1, 1\}$. As mentioned before, for the arc (v_i, v_j), $V_{v_i,v_j} = -1$ will indicate that the edge will be drawn in the lower half-plane and $V_{v_i,v_j} = +1$ in the upper one. $V_{v_i,v_j} = 0$ expresses the absence of edge.

Initially, the state of the net of vertices is randomly selected as a permutation of $\{1, 2, \ldots, N\}$. At any time, the net is looking for a better solution than the current one, in terms of minimizing the energy function.

In this paper, we study the permutation of two nodes and the change in the location of an edge. These produce the energy increment given in the next subsections.

3.1 Permutation of Two Nodes

When two vertices v_a and v_b permute their order a and b in the node line, we should take into account that the only edges changing their position (and therefore changing the number of crossings) are those that have exactly one endpoint in $\{v_a, v_b\}$.

Let us study the increase in the number of crossings depending on the relative positions of the endpoints.

Let us see how the number of crossings with the edge (v_x, v_a) is modified when it becomes the edge (v_x, v_b) since nodes a and b permute their positions. Hence, the arc represented with endpoints (x, a) will be drawn, after the update, with endpoints (x, b), and the only edges modifying the number of crossings due to the change must be in the same half-plane and must have an endpoint v_s represented between a and b ($a < s < b$) and the other, v_t, outside that interval ($(t < a) \vee (t > b)$). Some cases, depending on the position of x, are considered:

1. Case $x < a < s < b$: As shown in Fig. 3 (1), if $t < x < a < s < b$ the number of crossings is increased in one unit, since the edge (t_1, s) crosses the arc (x, b), but not (x, a). If $x < t < a < s < b$, a crossing disappears (the arc (t_2, s) cuts (x, a) but not (x, b)) and, at last, if $x < a < s < b < t$, the number of crossings will be increased in 1 unit (analize the arc (s, t_3)).

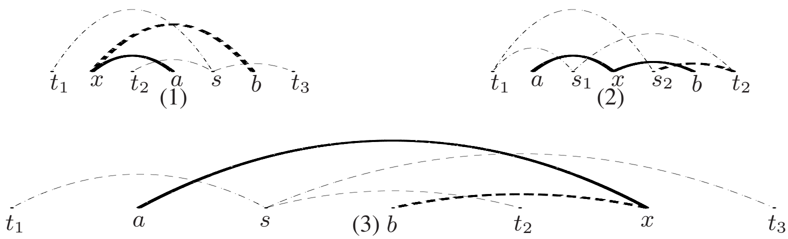

Fig. 3. Changes in the number of crossings when permuting nodes v_a and v_b, represented at positions a and b. An edge represented by the arc (a, x) will be transformed into the arc (b, x).

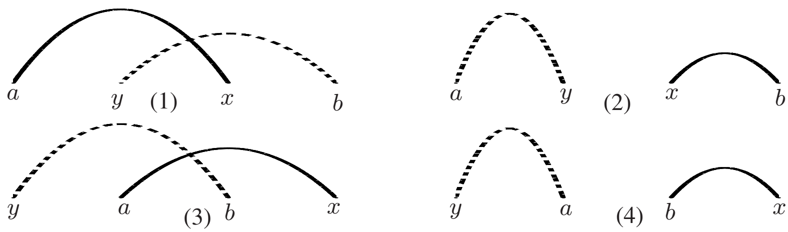

Fig. 4. Changes in the number of crossings when permuting nodes v_a and v_b. Edges represented by arcs (a, x) and (y, b) are transformed into arcs (b, x) and (y, a).

2. Case $a < x < b$: As shown in Fig. 3 (2), if $t < a < x < s < b$, or $a < x < s < b < t$, a new crossing is introduced (represented by the cuts of arcs (s_2, t_1) and (s_2, t_2) with the new edge (x, b)), whereas if $t < a < s < x < b$ or $a < s < x < b < t$ the number of crossings is reduced since crossings of (s_1, t_1) and (s_1, t_2) with (a, x) disappear).
3. Case $a < s < b < x$: A crossing is introduced if $a < s < b < t < x$ (arc (s, t_2)) and will be erased if $t < a < s < b < x$, or, $a < s < b < x < t$ (arcs (t_1, s) and (s, t_3)), as shown in Fig. 3 (3).

We must also take into account the change in the number of crossings with edges (v_x, v_b). Its study is similar to the already made for (v_x, v_a), it suffices to permute the literals a and b and to change the sense of the inequalities.

Finally, let us consider changes in the number of crossings produced between edges (v_a, v_x) and (v_b, v_y). All possible changes are shown in Fig. 4. There are different cases:

1. Case $a < y < x < b$:
   - Edges (a, x) and (y, b) (Fig. 4 (1)) are transformed into (x, b) and (y, a) (Fig. 4 (2)), vanishing the existing crossing.
   - Edges (a, y) and (x, b) (Fig. 4 (2)) are transformed into (b, y) and (x, a) (Fig. 4 (1)), causing the apparition of a crossing.
2. Case $y < a < b < x$:
   - When edges (a, x) and (y, b) (Fig. 4 (3)) are transformed into (x, b) and (y, a) (Fig. 4 (4)), a new crossing is formed.

– Arcs (a, y) and (x, b) (Fig. 4 (4)), are transformed into (b, y) and (x, a) (Fig. 4 (3)), and a crossing is eliminated.

We can derive an explicit formula for the increase of energy related to all these cases, just by considering that Eq. (2) (the number of crossings) can be rewritten as:

$$E = \sum_i \sum_j w_{i,j} \sum_k w_{j,k} \sum_l w_{k,l} \delta(V_{v_i,v_k}, V_{v_j,v_l})(1 - \delta(V_{v_i,v_k}, 0))$$

and by denoting $g(x, y, s, t) = \delta(V_{x,s}, V_{y,t})(1-\delta(V_{x,s}, 0))$, then the increase of energy caused by the permutation of nodes a and b is given by:

$$\begin{aligned} \Delta E = & \sum_{i\in\{a,b\}} \sum_j w_{i,j} \sum_k w_{j,k} \sum_l w_{k,l} \left(g(v_i, v_j, v_k, v_l) - g(v_i', v_j, v_k, v_l)\right) + \\ & + \sum_i \sum_{j\in\{a,b\}} w_{i,j} \sum_k w_{j,k} \sum_l w_{k,l} \left(g(v_i, v_j, v_k, v_l) - g(v_i, v_j', v_k, v_l)\right) + \\ & + \sum_i \sum_j w_{i,j} \sum_{k\in\{a,b\}} w_{j,k} \sum_l w_{k,l} \left(g(v_i, v_j, v_k, v_l) - g(v_i, v_j, v_k', v_l)\right) + \\ & + \sum_i \sum_j w_{i,j} \sum_k w_{j,k} \sum_{l\in\{a,b\}} w_{k,l} \left(g(v_i, v_j, v_k, v_l) - g(v_i, v_j, v_k, v_l')\right) \end{aligned} \tag{3}$$

where $v_s' = v_a$ if $v_s = v_b$; $v_s' = v_b$, if $v_s = v_a$; otherwise $v_s' = v_s$.

3.2 Change of the Position of an Edge

When the edge with endpoints v_a, v_b is represented in a given half-plane and its location changes (from the upper to the lower half-plane, or viceversa), an increase (or decrease) of the energy function (number of crossings) is produced and is given by $\Delta E =$

$$= (1 - \delta(V_{v_a,v_b}, 0)) \sum_{a<s<b} \sum_{(t<a)\vee(t>b)} (2\delta(V_{v_a,v_b}, V_{v_s,v_t}) - 1)(1 - \delta(V_{v_s,v_t}, 0)) \tag{4}$$

This expression can be obtained from Eq. (2), by simplifying the difference between the number of crossings before and after the possible change, since edges (v_x, v_y) with both endpoints located between a and b in the node line ($a < x < b$, $a < y < b$), or both placed outside the interval $[a, b]$, i. e., $x, y \notin [a, b]$, do not contribute to modify the number of crossings with (v_a, v_b), which is the changed edge.

4 Implementation for Two-Page Book Crossing Number Problem

Several dynamics can be used to solve this problem with our model, but we have chosen the following one due to its simplicity and efficiency:

1. Initialization: Given a graph with N nodes and M edges, a random feasible initial configuration $\boldsymbol{V}_0$ is selected for the location of the nodes. This initial state will be a permutation of the set of node indices $\{1, 2, \ldots, N\}$.

   For the edges set, the initial state vector $\boldsymbol{W}_0$ will be a random element of $\{-1, 1\}^M$. The output $V_{v_i,v_j} = -1$ means that the arc (i, j) will be represented in the lower half-plane, and if $V_{v_i,v_j} = 1$, the arc will be placed in the upper half-plane.

2. Two positions a and b in the node line are selected in all possible ways.

   Firstly, the net studies the increase of energy when vertices v_a and v_b are permuted by using Eq. (3). If the energy is reduced, the net permutes the vertices: $v_a(t+1) = v_b(t)$, $v_b(t+1) = v_a(t)$. Secondly, the net studies to change the half-plane in which the edge (v_a, v_b) is located. To this end, the increase of energy given by Eq. (4) is computed and if it is negative, the change is done. Thus, $V_{v_a,v_b}(t+1) = -V_{v_a,v_b}(t)$.
3. If all possible combinations of two positions have been considered and no change has been made, both networks have converged to a local minimum of the energy function (the cost function) and state vectors represent the obtained solution.

5 Simulation Results

In this Section we test the performance of our model and compare it with the neural heuristic proposed in [13] for a test set formed by graphs belonging to well-known graph families. Concretely:

– Complete graphs K_n, where all n nodes are interconnected (no self-connections).
– Circulant graphs $C_n(a_1, \ldots, a_k)$, where $0 < a_1 < \ldots < a_k < \frac{n+1}{2}$ is a graph with n vertices such that vertex i is adjacent to vertices $i \pm a_1, \ldots, i \pm a_k(\mathrm{mod}(n))$.

In order to compare the efficiency, we use the results given in [9] for Cimikowski's neural heuristic (CN) and for branch-and-bound algorithm (B&B). B&B results are only shown to give an idea of the optimal solutions.

Table 1. Comparative results of our proposed model with respect to Cimikowski's model and branch-and-bound heuristic for circulant graphs $C_n(.)$. A two-values entry (LB:UB) indicates lower and upper bounds for the optimal solution by B&B.

Graph	Prop.	CN	B&B	Graph	Prop.	CN	B&B
$C_{20}(1,2)$	0 (5.6)	2	0	$C_{30}(1,3,5,8)$	282 (342.9)	302	36:1980
$C_{20}(1,2,3)$	22 (29.6)	24	22	$C_{30}(1,2,4,5,7)$	406 (442.6)	398	66:3540
$C_{20}(1,2,3,4)$	74 (89.9)	74	26:870	$C_{32}(1,2,4,6)$	158 (224.4)	190	38:2256
$C_{22}(1,2)$	2 (6.7)	2	0	$C_{34}(1,3,5)$	115 (132.6)	106	6:1122
$C_{22}(1,2,3)$	22 (31.2)	26	24	$C_{34}(1,4,8,12)$	320 (360.3)	574	40:2550
$C_{22}(1,3,5,7)$	205 (229.6)	200	28:1056	$C_{36}(1,2,4)$	60 (83.2)	54	6:1260
$C_{24}(1,3)$	12 (17.0)	14	12	$C_{36}(1,3,5,7)$	349 (377.1)	328	42:2862
$C_{24}(1,3,5)$	76 (90.7)	78	72	$C_{38}(1,7)$	58 (67.4)	86	84
$C_{24}(1,3,5,7)$	239 (261.0)	220	30:1260	$C_{38}(1,4,7)$	20 (36.1)	192	6:1406
$C_{26}(1,3)$	11 (20.2)	16	14	$C_{40}(1,5)$	44 (62.8)	58	56
$C_{26}(1,3,5)$	78 (95.4)	82	6:650	$C_{42}(1,4)$	44 (62.5)	42	42
$C_{26}(1,4,7,9)$	341 (363.0)	364	32:1482	$C_{42}(1,3,6)$	131 (190.0)	160	6:1722
$C_{28}(1,3)$	16 (25.1)	16	14	$C_{42}(1,2,4,6)$	216 (311.2)	246	48:3906
$C_{28}(1,3,5)$	91 (106.5)	86	6:756	$C_{44}(1,4,5)$	119 (170.0)	180	6:1892
$C_{28}(1,2,3,4)$	114 (133.8)	110	34:1722	$C_{44}(1,4,7,10)$	567 (686.8)	638	50:4290
$C_{28}(1,3,5,7,9)$	589 (649.6)	560	62:3080	$C_{46}(1,4)$	46 (60.6)	46	46
$C_{30}(1,3,5)$	97 (109.9)	96	6:870	$C_{46}(1,5,8)$	299 (350.7)	296	6:2070

For every graph, 10 independent executions of our model were performed.

In the case of the complete graphs K_n, with $n \in \{5, \ldots, 13\}$, our model always has achieved the known theoretical optimal solution, as well as Cimikowski's model did.

For circulant graphs $C_n(.)$, comparative results are shown in Table 1. Values between parentheses are the average number of crossings among 10 executions.

It can be observed that, although average values of our model are not lower than the optimal solutions of Cimikowski's, our best solution is, in many cases, lower.

In addition, CN is not able to perform better than B&B when the optimal solution achieved by the latter is known, whereas our network improves it in some cases.

We must note that CN is a neural model which needs the fine-tuning of some parameters, and our model does not. Also, CN needs of a preprocessing step in which graph nodes are ordered and then remain fixed along the iterations. Our model is able to dynamically obtain a very good ordering.

6 Conclusions and Future Work

In this work we have presented a new neural model especially designed to solve some kinds of combinatorial optimization problems. This model is a variant of the multivalued model MREM formed by two networks. The dynamics of each of these networks depends on the outputs of the other network, and they are updated alternatively, to reach an equilibrium state corresponding to a local minimum of the common energy function.

We have tested our model with the well-known 2PLCN problem from graph theory. The proposed model avoids some of the drawbacks of other models in specialized literature, like the absence of convergence guarantees or the fine-tuning of parameters. In addition, it does not need a preprocessing stage to obtain a good node ordering, since it can be achieved dynamically.

By using some test instances, we have observed that our model is, at least, comparable to the other models, and it is able to achieve, in many cases, better solutions.

Future research lines cover aspects such as developing new dynamics for the model which could help to achieve better results. This model is also applicable when the node line is not a straight line, it can be a circle, or another geometry. Moreover, this model can also be used, with little modifications, to the k-pages book crossing problem, due to the ability of MREM to work with multivalued neuron outputs.

References

1. Chinn, P.Z., Chvátalová, L., Dewdney, A.K., Gibbs, N.E.: The bandwidth problem for graphs and matrices – a survey. J. Graph Theory 6, 223–253 (1982)
2. Kainen, P.C.: The book thickness of a graph, ii. Congr. Numer. 71, 127–132 (1990)
3. Chung, F.R.K., Leighton, F.T., Rosenberg, A.L.: Embedding graphs in books: a layout problem with applications to vlsi design. SIAM J. Alg. Disc. Meth. 8, 33–58 (1987)
4. Malitz, S.M.: On the page number of graphs. J. Algorithms 17, 71–84 (1994)
5. Ullman, J.D.: Computational Aspects of VLSI. Computer Science Press (1984)
6. Raghavan, R., Sahni, S.: Single row routing. IEEE Trans. Comput C-32, 209–220 (1983)
7. Sinden, F.W.: Topology of thin films circuit. Bell Syst. Tech. Jour. XLV, 1639–1666 (1966)

8. Tamassia, R., Di Battista, G., Batini, C.: Automatic graph drawing and readability of diagrams. IEEE Trans. Syst. Man. Cybern. SMC-18, 61–79 (1988)
9. Cimikowski, R.: Algorithms for the fixed linear crossing number problem. Discrete Applied Mathematics 122, 93–115 (2002)
10. Nicholson, T.A.J.: Permutation procedure for minimising the number of crossings in a network. In: Proc. IEE 115, 21–26 (1968)
11. Garey, M.R., Johnson, D.S.: Crossing number is np-complete. SIAM J. Alg. Disc. Methods 4, 312–316 (1983)
12. Masuda, S., Nakajima, K., Kashiwabara, T., Fujisawa, T.: Crossing minimization in linear embeddings of graphs. IEEE Trans. Comput. 39, 124–127 (1990)
13. Cimikowski, R., Shope, P.: A neural-network algorithm for a graph layout problem. IEEE Transactions on Neural Networks 7, 341–345 (1996)
14. Takefuji, Y.: Design of parallel distributed cauchy machines. In: Proc. Int. Joint Conf. Neural Networks, pp. 529 – 536 (1989)
15. Takefuji, Y.: Neural Network Parallel Computing. Kluwer Academic, Dordrecht (1992)
16. Mérida-Casermeiro, E., Galán-Marín, G., Muñoz Pérez, J.: An efficient multivalued hopfield network for the travelling salesman problem. Neural Processing Letters 14, 203–216 (2001)
17. Mérida-Casermeiro, E., Muñoz Pérez, J., Domínguez-Merino, E.: An n-parallel multivalued network: Applications to the travelling salesman problem. LNCS 2686, 406 – 413 (2003)
18. Mérida-Casermeiro, E., López-Rodríguez, D.: Graph partitioning via recurrent multivalued neural networks. Lecture Notes in Computer Science 3512, 1149 – 1156 (2005)
19. López-Rodríguez, D., Mérida-Casermeiro, E., Ortiz-de Lazcano-Lobato, J.M., López-Rubio, E.: Image compression by vector quantization with recurrent discrete networks. Lecture Notes in Computer Science 4132 (2006) 595–605