Drawing Graphs in Parallel Lines with Artificial Neural Networks

E. Mérida-Casermeiro, D. López-Rodríguez
Dept. of Applied Mathematics
ETSI Informática
University of Málaga, 29071 Málaga, Spain
{merida,dlopez}@ctima.uma.es

Abstract

In this work, we propose the use of a multivalued recurrent neural network with the aim of graph drawing. Particularly, the problem of drawing a graph in two parallel lines with the minimum number of crossings between edges is studied, and a formulation for this problem is presented. The neural model MREM is used to solve this problem. This model has been successfully applied to other optimization problems. In this case, a slightly different version is used, in which the neuron state is represented by a two dimensional discrete vector, representing the nodes assigned to a given position in each of the parallel lines. Some experimental simulations have been carried out in order to compare the efficiency of the neural network with a heuristic approach designed to solve the problem at hand. These simulations confirm that our neural model outperforms the heuristic approach, obtaining a lower number of crossings on average.

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" (where K half-planes or *pages* intersect) and then edges are added to this representation as specified by the adjacency matrix. The objective of this problem is to minimize the total number of crossings (adding over all K pages) produced by such layout.

Some examples of problems associated to linear graph representation are the bandwidth problem [1], the book thickness problem [8], the page-number problem [2, 11], the boundary VLSI layout problem [20], the single-row routing problem [17] and automated graph drawing [19]. Another important application is the design of printed circuit boards [18], since, for the case of 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 pages [3].

The problem studied in this work is a variant of the linear graph layout problem discussed above. In our case, graph nodes are distributed along two parallel straight lines, and edges can be drawn between nodes in the same line or in different lines.

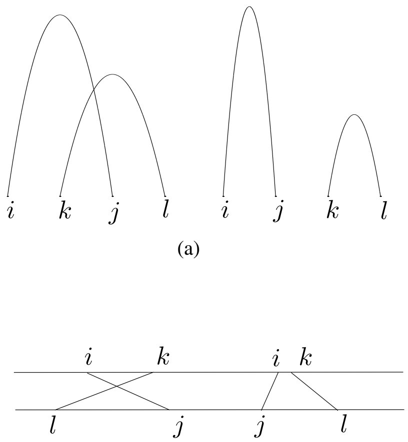
This problem has been studied in a different version by Giacomo *et al.* [4]. In that work, authors studied the particular case in which the graph to be represented was bipartite, although the representation could be done in two parallel convex curves (not necessarily straight lines).

In this work we present a neural model designed to solve this problem. The optimal node location is computed by the model, with the aim of minimizing the number of crossings that appear in the graphical representation. A previous version of this model was used to solve the K-pages crossing number problem [9].

Our model is a variant of the multivalued MREM model which has obtained very good results when applied to other combinatorial optimization problems [13, 15, 14, 10], guaranteeing the convergence to local minima of the energy function, identified with the objective function of the optimization problem.

The remainder of this work is structured as follows: in Sec. 2, the 2-Lines crossing problem is detailedly formulated. Later, in Sec. 3, the neural model used in this paper is described, as well as its application to solve the problem at hand. In Sec. 4, we show some experimental results for this problem. Finally, in Sec. 5, conclusions to this work are presented.

978-0-7695-3326-1/08 $25.00 © 2008 IEEE
DOI 10.1109/HIS.2008.89



(b)

Figure 1. Conditions on i, j, k and l to produce crossings: 1(a) $i < k < j < l$ and 1(b) $i < k$ and $l < j$.

2 Drawing Graphs in Parallel Lines

Let us consider a graph $G = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_i : i = 1, \ldots, N\}$ the set of nodes, and $\mathcal{E} = (e_{i,j})$ is the adjacency matrix, such that $e_{i,j} = 1$ if, and only if, edge (v_i, v_j) is present in the graph.

The objective of the 2-Lines Crossing Problem (2LCP) is to find the optimal location of graph nodes into two parallel straight lines minimizing the total number of crossings produced.

This problem is similar to the 1-page graph layout problem, since crossings between edges corresponding a nodes represented on the same line must be taken into account. The difference in this case is that crossings can be produced between edges with end-points in different lines. In Fig. 1, situations leading to crossings are graphically represented.

In order to count the number of crossings, we must first define some variables: Let $v_1(i)$ and $v_2(i)$ be the nodes located at position i in lines 1 and 2, respectively, for all i. For simplicity, and without loss of generality, assume that $i \in \{1, \ldots, N\}$. If no vertex is assigned to a location, let us denote $v_1(i) = N + 1$ or $v_2(i) = N + 1$, as needed.

The *augmented adjacency matrix* $\overline{\mathcal{E}}$ is defined as

$$\overline{\mathcal{E}} = \left(\begin{array}{c|c} \mathcal{E} & 0_{N\times 1} \\ \hline 0_{1\times N} & 0_{1\times 1} \end{array} \right)$$

to express that the "virtual" node $N + 1$ is not connected to the rest of vertices. For simplicity, let us denote the elements of this matrix as $e_{i,j}$.

The use of the virtual node $N + 1$ allows to change the number of nodes allocated in each of the parallel lines, if needed. Since the number of vertices in each line is not necessarily kept constant, the solution to the problem can be improved more easily.

Intra-line Crossings: Let us consider 4 positions in one of the node lines, verifying $1 \le i < k < j < l \le N$, where locations i, j, k and l are assigned to nodes $v_s(i)$, $v_s(j)$, $v_s(k)$ and $v_s(l)$. Then, edges $(v_s(i), v_s(j))$ and $(v_s(k), v_s(l))$ (if both exist) cross each other ($s \in \{1, 2\}$).

Thus, the total number of intra-line crossings is:

$$C_{\text{intra}} = \sum_{s\in\{1,2\}} \sum_{i=1}^{N} \sum_{k>i} \sum_{j>k} \sum_{l>j} e_{v_s(k),v_s(l)} e_{v_s(i),v_s(j)} \tag{1}$$

Inter-lines Crossings: A crossing is produced between edges $(v_1(i), v_2(j))$ and $(v_1(k), v_2(l))$ if, and only if, $i < k$ and $l < j$.

The number of crossings, produced by this kind of edges, is then:

$$C_{\text{inter}} = \sum_{i=1}^{N} \sum_{k>i} \sum_{j=1}^{N} \sum_{l<j} e_{v_1(k),v_2(l)} e_{v_1(i),v_2(j)} \tag{2}$$

The total number of crossings, produced by a given ordering of nodes in two lines, can be expressed in the following terms, by adding up Eqs. (1) and (2):

$$\begin{aligned} C &= C_{\text{intra}} + C_{\text{inter}} = \\ &= \sum_{s\in\{1,2\}} \sum_{i=1}^{N} \sum_{k>i} \sum_{j>k} \sum_{l>j} e_{v_s(k),v_s(l)} e_{v_s(i),v_s(j)} + \\ &\quad + \sum_{i=1}^{N} \sum_{k>i} \sum_{j=1}^{N} \sum_{l<j} e_{v_1(k),v_2(l)} e_{v_1(i),v_2(j)} \end{aligned}$$

3 Multivalued Neural Model for Graph Drawing

In this work, a multivalued recurrent neural network for optimization is applied to solve the 2LCP.

This neural model, MREM (Multivalued REcurrent Model) [12], 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 $\mathcal{M}$.

The state vector $\vec{V} = (V_1, V_2, \ldots, V_N) \in \mathcal{M}^N$ 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:

$$E(\vec{V}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{i,j} f(V_i, V_j) + \sum_{i=1}^{N} \theta_i(v_i) \qquad (3)$$

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.

Note that, due to these definitions, the MREM model is a generalization of Hopfield's models [6, 7] and other multivalued models, such as SOAR [16] and MAREN [5].

Multiple dynamics can be defined for this neural model. The usual deterministic dynamics consists in selecting a group of neurons and update them in order to decrease the value of the energy function.

In order to get 2LCP solved by MREM, we must identify a network state to a solution, that is, a vector state must represent the location of every node in each line.

To this end, define $V_i = (v_1(i), v_2(i)) \in \{1, \ldots, N, N+1\}^2$ the state of the i-th neuron, where $v_1(i)$ and $v_2(i)$ have the same meaning as in the previous section, i.e., $V_i = (n_1, n_2)$ means that nodes n_1 and n_2 are located on the i-th position on lines 1 and 2, respectively. If any of n_1 or n_2 is $N + 1$, then the respective location is empty. Note that, in this case, the possible states for a given neuron are represented by a 2-dimensional discrete vector. This is possible due to the use of the MREM network.

With this definition, the objective function to be minimized by the network is $C = C_{\text{intra}} + C_{\text{inter}}$, as defined above.

This objective function can be rewritten as an energy function of the MREM model (Eq. (3)). In order to do so, define

$$\rho_{i,j} = \begin{cases} 1, & \text{if } j > i \\ 0, & \text{otherwise} \end{cases}$$

Then, we arrive at:

$$C = \sum_i \sum_j \sum_k \sum_\ell (\rho_{i,k}\rho_{\ell,j} e_{v_1(k),v_2(\ell)} e_{v_1(i),v_2(j)} + \\ +\rho_{i,k}\rho_{k,j}\rho_{j,\ell} \sum_{s\in\{1,2\}} (e_{v_s(k),v_s(\ell)} e_{v_s(i),v_s(j)}))$$

From this, we can deduce expressions for:

$$w_{i,j} = -2$$

$$f(V_i, V_j) = \sum_k \sum_\ell (\rho_{i,k}\rho_{\ell,j} e_{v_1(k),v_2(\ell)} e_{v_1(i),v_2(j)} + \\ +\rho_{i,k}\rho_{k,j}\rho_{j,\ell} \sum_{s\in\{1,2\}} (e_{v_s(k),v_s(\ell)} e_{v_s(i),v_s(j)}))$$

and $\theta_i \equiv 0$, for all $i, j \in \{1, \ldots, N\}$.

The computational dynamics of the neural network to solve the problem at hand consists on permuting the location of two nodes (including the virtual $N + 1$ node) with a random feasible initial configuration of the network. That is, select (sequentially) $s_1, s_2 \in \{1, 2\}$ and $i, j \in \{1, \ldots, N\}$ with $v_{s_1}(i) = n_1$ and $v_{s_2}(j) = n_2$. Then, the network studies the increase or decrease in the number of crossings in the case of making $v_{s_1}(i) = n_2$ and $v_{s_2}(j) = n_1$. If the number of crossings is reduced, then the update is done.

Note that there are two possible cases (the trivial case in which $v_{s_1}(i) = v_{s_2}(j) = N + 1$ is excluded):

- Both $v_{s_1}(i), v_{s_2}(j) \leq N$. Then, the network studies the permutation of the location of these two nodes.

- Only one of the nodes is the virtual node $N + 1$. Then, this dynamics is equivalent to change one node location. In this case, the number of nodes assigned to each of the lines may change, allowing to find a better solution.

Observe that, with this dynamics, the state of the network remains feasible along the iterations.

4 Experimental Results

In this Section we test the performance of our model and compare it with a heuristic method, developed *ad hoc* for this problem, for a test set formed by graphs belonging to well-known graph families. Concretely:

- Graphs $K_{n,m}$, formed by two groups of nodes, one of cardinality n and another with m. Each node of a group is adjacent to every node in the other group.

- Circulant graph $C_n(a_1, \ldots, a_k)$, 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 $i \pm a_1, \ldots, i \pm a_k(\text{mod}(n))$. The circulant graph $C_n(a_1, \ldots, a_k)$ has $n \cdot k$ edges.

- Hipercubic graphs H_n, with 2^n vertices, which are numbered (in base 2) from 0 to $2^n - 1$ by using n bits. There exist the edge (v_i, v_j) if, and only if, the binary representations of these two nodes differ in just one bit. This definition can be explained as follows: vertices are the corners of a hypercube in dimension n, and edges join neighboring corners.

The heuristic method used to compare results consists in sequentially assign a location (in one of the 2 lines) to each node (sorted according to a given criterion), such that the number of crossings produced by inserting the node in the representation is minimum. In order to produce better

Graph	MREM		Heur1	Heur2
	Min	Av.		
$K_{5,5}$	16	16.24	40	40
$K_{5,7}$	36	36.34	90	84
$K_{5,9}$	64	64.28	160	144
$K_{5,11}$	100	100.68	250	220
$K_{5,13}$	144	144.96	360	312
$K_{7,7}$	81	81	189	189
$K_{7,9}$	144	144.24	336	324
$K_{7,11}$	225	225.36	525	495
$K_{7,13}$	324	324	756	702
$K_{9,9}$	256	256.48	576	576
$K_{9,11}$	400	400.24	900	880
$K_{9,13}$	576	576.24	1296	1248
$K_{11,11}$	625	625.4	1375	1375
$K_{11,13}$	900	900.4	1980	1950
$K_{13,13}$	1296	1296	2808	2808

Table 1. Results of the techniques proposed in this work for $K_{n,m}$.

Graph	MREM		Heur1	Heur2
	Min	Av.		
H_3	0	0.76	5	5
H_4	8	10.55	29	29
H_5	74	96.16	168	168
H_6	478	630.2	895	895

Table 2. Results of the techniques proposed in this work for the hipercubic graphs H_n.

results, two strategies for sorting nodes have been considered: nodes can be sorted in decreasing (Heur1) or increasing (Heur2) order of degree.

That is, given nodes $v_{\sigma(1)}, \ldots, v_{\sigma(N)}$ (sorted according to a permutation σ), first, assign node $v_{\sigma(1)}$ to line 1. At step n, assign node $v_{\sigma(n)}$ to the line in which its insertion produces a lower number of crossings, taking into account only edges corresponding to nodes already assigned.

For each of the test graphs, 100 independent runs of our neural model have been studied. Note that the heuristic approaches mentioned before always achieve the same solution. However, the MREM model, since its initial state is randomly selected, may achieve better solutions.

The results of applying these techniques are shown in Tables 1, 2 and 3.

In those tables, the minimum and average number of crossings obtained in 100 runs of the neural model MREM are shown, as well as the result of applying the heuristic approach mentioned before, in its two variants, Heur1 and Heur2. It can be observed that our neural model is able to outperform the proposed heuristic, since the average number of crossings achieved by MREM is, in most cases, lower than the obtained by the heuristic. Note that results for both heuristic strategies are the same. There is no difference between both variants. However, experiments have been performed with the heuristic algorithm in which nodes were initially randomly sorted. In those experiments, the number of crossings was much higher than with Heur1 and Heur2 (about 2 or 3 times higher).

A graphical representation of a solution for $K_{3,5}$, with 4 crossings, is shown in Fig. 2.

Graph	MREM		Heur1	Heur2
	Min	Av.		
$C_{22}(1,2)$	0	4.53	2	2
$C_{22}(1,2,3)$	24	32.89	30	30
$C_{22}(1,3,5,7)$	187	209.95	304	304
$C_{24}(1,3)$	9	14.4	24	24
$C_{24}(1,3,5)$	65	82.69	133	133
$C_{24}(1,3,5,7)$	200	230.42	328	328
$C_{26}(1,3)$	11	16.96	20	20
$C_{26}(1,3,5)$	72	89.88	134	134
$C_{26}(1,4,7,9)$	307	334.75	518	518
$C_{28}(1,3)$	12	18.84	24	24
$C_{28}(1,3,5)$	78	100.63	129	129
$C_{28}(1,2,3,4)$	110	142.79	185	185
$C_{28}(1,3,5,7,9)$	536	591.62	850	850
$C_{30}(1,3,5)$	86	107.84	135	135
$C_{30}(1,3,5,8)$	265	325.27	431	431
$C_{30}(1,2,4,5,7)$	423	461.22	635	635
$C_{32}(1,2,4,6)$	144	201.69	244	244
$C_{34}(1,3,5)$	103	127.11	154	154
$C_{34}(1,4,8,12)$	299	365.51	834	834
$C_{36}(1,2,4)$	45	83.41	82	82
$C_{36}(1,3,5,7)$	303	386.13	511	511
$C_{38}(1,7)$	42	59.9	106	106
$C_{38}(1,4,7)$	165	206.3	231	231
$C_{40}(1,5)$	34	55.59	74	74
$C_{42}(1,4)$	28	52.91	42	42
$C_{42}(1,3,6)$	122	171.06	223	223
$C_{42}(1,2,4,6)$	199	304.67	314	314
$C_{44}(1,4,5)$	115	182.93	172	172
$C_{44}(1,4,7,10)$	535	655.58	801	801
$C_{46}(1,4)$	33	57.76	48	48

Table 3. Results of the techniques proposed in this work for circulant graphs $C_n(a_1, \ldots, a_k)$.

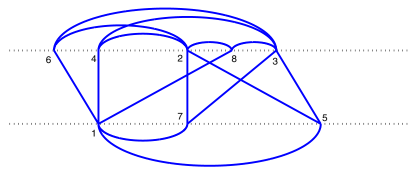


Figure 2. Result of drawing graph $K_{3,5}$ in two parallel lines.

5 Conclusions and Future Work

In this paper, the neural model MREM is used to solve the problem of drawing a graph in two parallel lines with the minimum number of crossings. The mathematical formulation for this problem is also presented.

A new feature has been introduced in the neural model proposed in this paper: the state of the i-th neuron is a two dimensional vector representing the nodes located at the i-th position in each of the parallel lines.

Experimental results show that MREM is able to outperform a heuristic approach designed *ad hoc* for solving the problem at hand.

Further improvements are expected with a parallel implementation of the neural network (observe that in this paper, the dynamics of the network is sequential).

In addition, a generalization to the case in which there are $K > 2$ parallel lines is under study, as well as other possible layouts for graph drawing.

Acknowledgments

This work is partially supported by Junta de Andalucía (Spain) under contract TIC-01615, project name Intelligent Remote Sensing Systems.

References

[1] P. Z. Chinn, L. Chvátalová, A. K. Dewdney, and N. E. Gibbs. The bandwidth problem for graphs and matrices – a survey. *J. Graph Theory*, 6:223 – 253, 1982.

[2] F. R. K. Chung, F. T. Leighton, and A. L. Rosenberg. Embedding graphs in books: a layout problem with applications to vlsi design. *SIAM J. Alg. Disc. Meth.*, 8:33 – 58, 1987.

[3] R. Cimikowski. Algorithms for the fixed linear crossing number problem. *Discrete Applied Mathematics*, 122:93 – 115, 2002.

[4] E. Di Giacomo, L. Grilli, and G. Liotta. Drawing bipartite graphs on two curves. *Lecture Notes in Computer Science*, 4372:380 – 385, 2007.

[5] M. H. Erdem and Y. Ozturk. A new family of multivalued networks. *Neural Networks*, 9(6):979 – 989, 1996.

[6] J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. volume 79, pages 2254 – 2558, 1982.

[7] J. Hopfield and D. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52:141 – 152, 1985.

[8] P. C. Kainen. The book thickness of a graph, ii. *Congr. Numer.*, 71:127 – 132, 1990.

[9] D. López-Rodríguez, E. Mérida-Casermeiro, J. Ortiz-de Lazcano-Lobato, and G. Galán-Marín. k-pages graph drawing with multivalued neural networks. *Lecture Notes in Computer Science*, 4669:816 – 825, 2007.

[10] D. López-Rodríguez, E. Mérida-Casermeiro, J. M. Ortiz-de Lazcano-Lobato, and E. López-Rubio. Image compression by vector quantization with recurrent discrete networks. *Lecture Notes in Computer Science*, 4132:595 – 605, 2006.

[11] S. M. Malitz. On the page number of graphs. *J. Algorithms*, 17(1):71 – 84, 1994.

[12] E. Mérida-Casermeiro. *Red Neuronal recurrente multivaluada para el reconocimiento de patrones y la optimización combinatoria*. PhD thesis, Universidad de Málaga, 2000.

[13] E. Mérida-Casermeiro, G. Galán-Marín, and J. Muñoz Pérez. An efficient multivalued hopfield network for the travelling salesman problem. *Neural Processing Letters*, 14:203 – 216, 2001.

[14] E. Mérida-Casermeiro and D. López-Rodríguez. Graph partitioning via recurrent multivalued neural networks. *Lecture Notes in Computer Science*, 3512:1149 – 1156, 2005.

[15] E. Mérida-Casermeiro, J. Muñoz Pérez, and E. Domínguez-Merino. An n-parallel multivalued network: Applications to the travelling salesman problem. *Lecture Notes in Computer Science*, 2686:406 – 413, 2003.

[16] Y. Ozturk and H. Abut. System of associative relationships (soar). 1997.

[17] R. Raghavan and S. Sahni. Single row routing. *IEEE Trans. Comput*, C-32(3):209 – 220, 1983.

[18] F. W. Sinden. Topology of thin films circuit. *Bell Syst. Tech. Jour.*, XLV:1639 – 1666, 1966.

[19] R. Tamassia, G. Di Battista, and C. Batini. Automatic graph drawing and readability of diagrams. *IEEE Trans. Syst. Man. Cybern.*, SMC-18:61 – 79, 1988.

[20] J. D. Ullman. *Computational Aspects of VLSI*. Computer Science Press, 1984.