Neural Networks for Optimization Problems in Graph Theory

Jenn-Shiang Lai and Sy-Yen Kuo
Dept. of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.

Ing-Yi Chen
Dept. of Electronic Engineering
Chung Yuan Christian University
Chungli, Taiwan, R.O.C.

Abstract

This paper presents a novel technique to map the minimum vertex cover and related problems onto the Hopfield neural networks. The proposed approach can be used to find near-optimum solutions for these problems in parallel, and particularly the network algorithm always yields minimal vertex covers. Further, the relationships between Boolean equations and arithmetic functions are presented. Based on these relationships, other NP-complete problems in graph theory can also be solved by neural networks. Extensive simulation was performed and the experimental results demonstrate that the network algorithm outperforms the well-known greedy algorithm for the vertex cover problem.

1 Introduction

The vertex cover problem is to find the smallest set of vertices that covers all the edges in a given graph. This is a very practical problem [1]. Since this problem is NP-complete, several sequential approximation algorithms were proposed [2]. However, these algorithms make decisions based solely on local information, and they may fail in many situations [1, 3]. In this paper, we propose an algorithm based on the neural network to solve this problem by Hopfield model [4].

The rest of this paper is organised as follows. In Section 2, we describe the technique to solve the vertex cover problem by the Hopfield network. Sections 3 and 4 derive approaches solving the maximum independent set and maximum clique problems, respectively. Section 5 shows experimental results for the vertex cover problem. Finally, conclusions are given in Section 6.

2 Vertex Cover Problem

Let $G = (N, A)$ be an undirected graph, where N is the set of vertices, A is the set of edges, and $|N|$ denote the number of vertices. The vertex cover problem is a problem of finding the smallest subset $C \subseteq N$ such that for each edge $[i, j] \in A$ at least one of i and j belongs to C [5]. Since the goal of this problem is to cover all edges of G with as few vertices as possible, selecting each time the single vertex that by itself covers as many of the remaining edges as possible is an attractive strategy. Henceforth, the well-known greedy algorithm for this problem is described as follows: successively select the vertex of largest degree (i.e., adjacent to the largest number of edges) and remove this vertex together with all of its adjacent edges from the graph until all edges have been removed. Although, the removed set of vertices is expected to be a minimum cover, it is easy to find some situations where this approximation fails to yield a minimum cover. Consider how this scheme be applied to graph of Figure 1 [3]. We first choose vertex I, and then II, III, IV, and V. The resulting vertex cover consists of 5 vertices. But the optimum vertex cover, $\{II, III, IV, V\}$, has just 4 vertices. This is because vertex I becomes redundant after selecting vertices II, III, IV and V, but this sequential algorithm can't remove any redundant vertex in a cover.

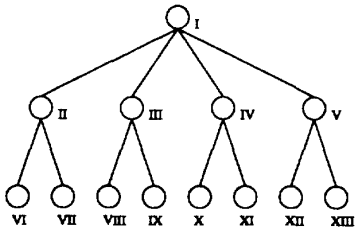

Figure 1: An example for the vertex cover problem.

For every undirected graph $G = (N, A)$, one can find a Hopfield model such that there is a one to one correspondence between the minima of the network and the minimal vertex covers of that graph. Let C be a vertex cover and the state V_i of neuron i be determined by

$$V_i = \begin{cases} 1, & \text{if vertex } i \text{ is in the cover } C \\ 0, & \text{otherwise} \end{cases}$$

Hence, if (a_{ij}) is the adjacency matrix of graph G, the vertex cover problem can be mathematically stated as

finding the minimum of the following cost function

$$E_C = \frac{1}{2}\left[\sum_i \sum_j a_{ij}\overline{V_i \vee V_j}\right] + \left[\sum_j \gamma V_j\right]$$

where $\vee$ is the logical OR, $\overline{X}$ means the complement of X, and $0 < \gamma < 1$.

The first expression in bracket goes to 0 when all edges are covered by C, and the second bracketed expression is used to minimise the number of vertices in C.

As shown in Table 1, the Boolean equation of logical variables can be represented by its corresponding arithmetic function and therefore, the former objective function can be expanded and rearranged as

$$\begin{aligned} E_C &= \frac{1}{2}\sum_i \sum_{j\neq i} a_{ij}\overline{V_i \vee V_j} + \sum_j \gamma V_j \quad (\text{since } a_{ii} = 0) \\ &= \frac{1}{2}\sum_i \sum_{j\neq i} a_{ij}\left[1 - (V_i + V_j - V_iV_j)\right] + \sum_j \gamma V_j \\ &= -\frac{1}{2}\sum_i \sum_{j\neq i} (-a_{ij}) V_iV_j \\ &\quad - \sum_j (\sum_i a_{ij} - \gamma)V_j + \frac{1}{2}\sum_i \sum_{j\neq i} a_{ij} \end{aligned}$$

This is in the form of the above Lyapunov function. Hence we can obtain an algorithm based on the Hopfield network with the external input $I_j = \sum_i a_{ij} - \gamma$ to neuron j, and the connection weight $w_{ij} = -a_{ij}$ between the i-th and the j-th neurons.

Table 1: Boolean and arithmetic representations for three typical functions.

Logic	Boolean	Arithmetic
NOT X	$\overline{X}$	1-X
X AND Y	$X \wedge Y$	XY
X OR Y	$X \vee Y$	$X + Y - XY$

If the initial states of all neurons are set to be randomly-generated values around 0.5 (say, 0.5 ± 0.05), and the continuous model of the Hopfield network is applied, this neural network approach will stabilize into a minimal vertex cover in parallel. On the other hand, let the state of every neuron be zero initially, by using the fast gradient-descent technique for the discrete Hopfield model (i.e., by sequentially updating the state of a neuron which can reduce the greatest amount of system energy), this network method is converged to find a cover with minimal number of vertices at all times.

Indeed, the fast gradient-descent network method works like the traditional greedy algorithm, so it is a $(\ln n)$-approximation algorithm [1]. Notice that the greedy algorithm can be considered as the fast gradient-descent algorithm with $\gamma = 0$; therefore, if the optimum solution consists of n vertices, the minimal vertex cover obtained by this algorithm may grow as fast as $(n + n\ln n)$. In practice, this sequential network can obtain the minimum cover of the graph in Figure 1, but this approach is very hard to find the optimum solution for the graph in Figure 2. To avoid this situation, the

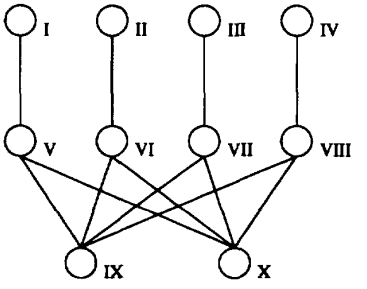

Figure 2: A 10-vertex 12-edge graph.

discrete network algorithm is adjusted to emulate the continuous one, and the objective function is modified into

$$E_C = \frac{1}{2}\left[\sum_i \sum_j a_{ij}\overline{V_i \vee V_j}\right] + \left[\sum_j \gamma_j V_j\right]$$

where

$$\gamma_j = 1 - \frac{1 + \sum_{i\neq j} a_{ij}}{|N|^2} = \left[1 - \frac{1}{|N|^2}\right] + \frac{\sum_{i\neq j} a_{ij}}{|N|^2}$$

In this way, when two vertices have the same degree in the remaining graph, the one with higher original degree will be selected (since it can cover more edges and may result in more redundancies being removed). Also, the external input to the j-th neuron should become

$$I_j = \left[1 + \frac{1}{|V|^2}\right]\sum_{i\neq j} a_{ij} - \left[1 - \frac{1}{|V|^2}\right]$$

and the worst cases in Figure 2 can be elegantly solved. Likewise, the first expression in bracket goes to 0 when all edges in G are covered, so the energy function of a minimal vertex cover C can be deduced into

$$\begin{aligned} E_C &= \sum_{i\in C} \gamma_i = \sum_{i\in C}\left[1 - \frac{1 + \deg(i)}{|N|^2}\right] \\ &= |C|\left[1 - \frac{1}{|N|^2}\right] - \frac{1}{|N|^2}\sum_{i\in C} \deg(i) \end{aligned}$$

and the larger the minimal vertex cover is, the higher its energy will be (see the following Lemma).

Lemma. Let the energy function of a minimal cover C of a graph $G = (N, A)$ be defined as

$$E_C = |C|\left[1 - \frac{1}{|N|^2}\right] - \frac{1}{|N|^2}\sum_{i \in C} \deg(i)$$

If C_1, C_2 are two minimal vertex covers, and $|C_1| > |C_2|$, then $E_{C_1} > E_{C_2}$.

Proof of lemma. Assume that χ is a positive integer and $|C_1| = |C_2| + \chi$, then $E_{C_1} - E_{C_2} = \chi(1 - \frac{1}{|N|^2}) - \frac{1}{|N|^2}\left[\sum_{i \in C_1^-} \deg(i) - \sum_{j \in C_2^-} \deg(j)\right]$, where $C_1^- = C_1 - (C_1 \cap C_2), C_2^- = C_2 - (C_1 \cap C_2)$.

After removing the edges not incident with C_1^- or C_2^- and those between C_1^- and C_2^-, we can obtain a graph H without changing $E_{C_1} - E_{C_2}$ (see Figure 3). In this way, any edge in H is incident with a vertex in $C_1 \cap C_2$ and another vertex in C_1^- or C_2^-. Hence,

$$0 \le \sum_{i \in C_j^-} \deg(i) \le |C_j^-||C_1 \cap C_2| \quad (j = 1, 2)$$

and

$$\begin{aligned} &\sum_{i \in C_1^-} \deg(i) - \sum_{j \in C_2^-} \deg(j) \\ &\le |C_1^-|(|N| - |C_1^-| - |C_2^-|) \\ &\le |C_1^-|(|N| - 2|C_1^-| + \chi) \le \frac{(|N| + \chi)^2}{8} \end{aligned}$$

Therefore,

$$E_{C_1} - E_{C_2} \ge \chi\left[1 - \frac{1}{|N|^2}\right] - \frac{(|N| + \chi)^2}{8|N|^2} > 0$$

□

3 Maximum Independent Set Problem

In this section, we are to derive a similar technique for solving the maximum independent set problem.

Given a graph $G = (N, A)$, the aim of the maximum independent set problem is to find the largest set S of vertices such that no two vertices in S are connected by an edge. It is well-known that if S is an independent set of G, $N - S$ is a vertex cover of G [5]. Further, $N - S$ is the optimal solution to the vertex cover problem iff S is the maximal independent set of G. So, the objective function of the maximum independent set problem can be formulated as

$$E_S = \frac{1}{2}\left[\sum_i \sum_j a_{ij}\overline{\overline{V_i} \vee \overline{V_j}}\right] + \left[\sum_j \gamma_j \overline{V_j}\right]$$

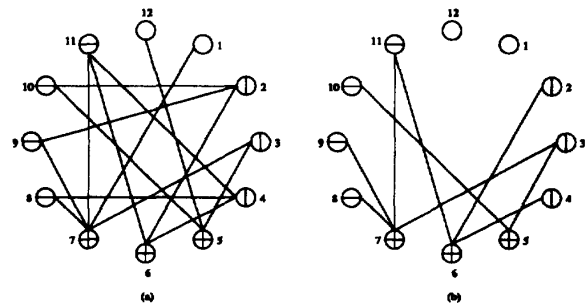

Figure 3: (a) A graph G with $C_1 = \{5, 6, 7, \ldots, 11\}$ and $C_2 = \{2, 3, 4, \ldots, 7\}$. (b) The corresponding graph H without changing $E_{C_1} - E_{C_2}$.

$$\begin{aligned} &= \frac{1}{2}\sum_i \sum_{j \ne i} a_{ij}(V_i \wedge V_j) + \sum_j \gamma_j \overline{V_j} \\ &= -\frac{1}{2}\sum_i \sum_{j \ne i}(-a_{ij})V_i V_j - \sum_j \gamma_j V_j + \sum_j \gamma_j \end{aligned}$$

and we can have the neural network algorithm to solve the maximum independent set problem.

4 Maximum Clique Problem

By definition, for a graph $G = (N, A)$ a maximal complete subgraph is called a clique, and the complement of the graph G is the graph $\overline{G}$ by deleting the edges of G from the complete graph on the same vertices. Accordingly, if (a_{ij}) and (b_{ij}) are the adjacency matrices of G and $\overline{G}$, respectively, we can have $b_{ij} = 1 - a_{ij}$ for $i \ne j$, and $b_{ii} = a_{ii} = 0$ for all i. Since the independent sets and cliques have the following relationships

1. K is a clique of G.
2. K is an independent set of $\overline{G}$.

it is easy to get the objective function of the maximum clique problem as

$$\begin{aligned} E_K &= \frac{1}{2}\left[\sum_i \sum_j b_{ij}\overline{\overline{V_i} \vee \overline{V_j}}\right] + \left[\sum_j \lambda_j \overline{V_j}\right] \\ &= -\frac{1}{2}\sum_i \sum_{j \ne i}(a_{ij} - 1)V_i V_j - \sum_j \lambda_j V_j + \sum_j \lambda_j \end{aligned}$$

and we can obtain a similar algorithm for the maximum clique problem.

5 Experimental Results

The algorithms are implemented in C on Sun SPARCstation 2 for graphs with edge density 5%, 10%, 25%,

and 50%. Here the edge density is the probability that an edge exists between a pair of vertices [3]. For each size, ten random graphs are examined and all algorithms were run for 25 times to find the optimum solution of every instance. In this simulation, all initial states for discrete techniques, such as the greedy algorithm in [2] or the fast gradient-descent networks ($\gamma = 1/2$ or $\gamma = (1 + \sum_{i \neq j} a_{ij})/|N|^2$), are zeros. In the continuous model, γ is set to $1/3$. Table 2 shows the probabilities of finding the minimum vertex covers by these algorithms. Here the sequential (greedy) algorithm is labelled as SA, the fast gradient-descent networks with $\gamma = 0.5$ and $\gamma = (\sum_{i \neq j} a_{ij})/|N|^2$ are denoted by FGDN1 and FGDN2, respectively, and the continuous Hopfield approach with $\gamma = 1/3$ is named CHA. Compared with the sequential algorithm, both gradient-descent networks have higher probabilities to find the optimum solutions in most cases.

Table 2: Probabilities of the minimum vertex covers.

Sizes(05%)	SA	FGDN1	FGDN2	CHA
10	0.9	0.912	1.0	0.996
20	0.848	0.828	0.9	0.892
30	0.568	0.648	0.584	0.608
40	0.48	0.58	0.688	0.74
50	0.36	0.396	0.452	0.404
Sizes(10%)	SA	FGDN1	FGDN2	CHA
10	0.98	0.98	1.0	0.98
20	0.56	0.66	0.588	0.488
30	0.552	0.604	0.664	0.556
40	0.508	0.572	0.72	0.744
50	0.288	0.372	0.548	0.428
Sizes(25%)	SA	FGDN1	FGDN2	CHA
10	0.904	0.996	1.0	0.976
20	0.752	0.832	0.884	0.76
30	0.552	0.66	0.772	0.748
40	0.448	0.536	0.508	0.496
50	0.404	0.476	0.512	0.48
Sizes(50%)	SA	FGDN1	FGDN2	CHA
10	0.936	0.948	0.916	0.908
20	0.904	0.924	0.9	0.968
30	0.856	0.884	1.0	0.828
40	0.796	0.848	0.9	0.952
50	0.832	0.856	0.868	0.764

6 Conclusions

In this paper, we have presented a novel method to derive the minimum vertex cover and its companions (maximum independent set and maximum clique problems) by neural networks. The proposed approach can be used to find good solutions for vertex cover problems in parallel, and the neural networks always converge to irredundant vertex covers of the given graphs.

The relationship between Boolean equations and arithmetic functions was also proposed. In addition to the problems discussed in this article, other NP-complete problems in graph theory can also be mapped onto the Hopfield neural networks with the same method. For instance, the bipartite subgraph problem and the graph partitioning of an n-vertex graph [6, 7].

A large number of simulations was performed to evaluate and justify our algorithm. Experimental results show that the performance of our method is better than that of the well-known sequential greedy algorithm, and due to the inherent properties of Hopfield neural networks, this algorithm is suitable for massively parallel execution. Moreover, with the advances in VLSI technology, large-scale neural networks may become available and this method will provide a significant advantage over others.

References

[1] C. H. Papadimitriou, and K. Steiglitz, "Combinatorial Optimization: Algorithms and Complexity," Prentice-Hall, Englewood Cliffs, N. J., pp. 358-409, 1982.

[2] D. Johnson, "Approximation algorithms for combinatorial problems," *J. Comput. Syst. Sci.*, Vol. 9, 1974.

[3] Y. Peng, J. A. Reggia, and T. Li, "A connectionist approach to vertex covering problems," *Int. J. Neural Syst.*, Vol. 3, No. 1, pp. 43-56, 1992.

[4] J. J. Hopfield, "Neural networks and physical system with convergent collective computational properties," *Proc. Nat. Acad. Sci. U.S.*, Vol. 79, pp. 2554-2558, 1982.

[5] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-completeness," San Francisco: Freeman, 1979.

[6] J. S. Lai and S. Y. Kuo, "A Hopfield network algorithm for the bipartite subgraph problem", 3rd International Conference for Young Computer Scientists, July, 1993.

[7] D. E. Van den Bout and T. K. Miller, III, "Graph partitioning using annealed neural networks," *IEEE Trans. on Neural Networks*, Vol. 1, No. 2, pp. 192-203, June 1990.