A Note on the Computational Complexity of Symmetric Connectionist Networks

Saibal Banerjee
Simon Kasif

Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218

January, 1988
Report JHU-88/1
A Note on the Computational Complexity of Symmetric Connectionist Networks

Saibal Banerjee
Simon Kasif
Department of Computer Science
The Johns Hopkins University
Baltimore, MD 21218

ABSTRACT

Connectionist models are currently being actively investigated by many researchers in Artificial Intelligence, Information Theory and Neurology. These networks have been shown to be applicable to a wide range of domains such as content addressable memories, semantic nets, computer vision, natural language parsing, speech and approximation schemes for hard optimization problems. A major problem that has not been addressed so far is the computational complexity of connectionist network algorithms. This question has two main aspects.

1. Determining the class of computations performed particularly well by these networks and the class of computations that are inherently unsuitable for connectionist implementation. Intuitively, given a problem \( P \) of input size \( N \), we would like to estimate whether a network of size \( S(N) \) can solve the problem in time \( T(N) \).

2. Determining the computational complexity of the optimization process performed by the network. Specifically, given a description of the network of size \( N \), can we find subexponential algorithms to find stable (and maximally stable) states in the network that correspond to local and global energy minima.

In this note we outline several approaches to the study of computational complexity for general connectionist networks as well as networks with given structural properties such as positive weight, bounded degree and sparse networks.
nodes in each layer (width). Connectivity and other graph theoretical properties of the network may play an important role as well. These and other questions will be addressed in this note.

2. Connectionist Networks

Connectionist models are currently being actively investigated by many researchers in Artificial Intelligence, Information Theory and Neurology. Though the roots of these models can be traced to the beginning of the century and the study of perceptrons [MP], the recent investigations have been inspired by a new approach for modeling neural networks advocated by Hopfield [Ho]. Hopfield [Ho], has postulated a model for neural organizations which has emergent collective computational abilities. The proposed model differs from previous ones because it takes into account the fact that neural networks can act synergistically, with parallel feedback and connectivity to greatly enhance their computational power (see [RM] for an extended list of references and a survey on the subject). This model has led to the development of the concept of a connectionist network.

Informally, a connectionist network is a large collection of computational units (nodes) with a finite number of states and with elementary computational capabilities. Information exchange between the various nodes in the network is supported by interconnections which are able to pass information in either direction. By such interconnections, every node can report its current state to every other node connected to it. The change of state of a computational unit is referred to as firing. Every node when firing, computes a weighted sum over all the nodes that it is connected to, the weights being arbitrarily negative or positive real numbers, and changes its state in accordance with a threshold rule. A cost or energy function can be used to describe the behavior of such a system of interacting nodes. According to the proposed model, a computation in a neural network proceeds by each node asynchronously updating its state. The computation stops when the energy of the system achieves a local minimum, so that at this particular stage if any one of the nodes changes its corresponding state, the total energy of the system increases.

Hopfield shows that this model of neural networks does exhibit collective computational properties. Each node is considered to be in one of two possible states, "on" or "off". Then a set of \( n \) distinguishable nodes can be thought of as a binary word of length \( n \). Hopfield proposed that the information storage points of neural networks are exactly those binary words which correspond to the local minima of the energy function of the system. Thus if the system is started with an initial configuration in which every node is at a state "close" enough to a local minimum configuration, it will immediately "plummet" down to that local minimum, by each node asynchronously updating its present state. This is supposed to be the basis of content-addressable memory in neural networks. It has been found that such a model can store information with stability and efficiency, and can retrieve that information with some error-correcting capability. The model is also believed to be quite robust, and should work even when more neurological details are added.

The Hopfield model of neural networks is closely related to various problems in a variety of disciplines. In Optimization theory, various problems including the Traveling Salesman can be modeled along these lines. In Artificial Intelligence, knowledge
representation schemes and learning systems are based on this model. In Physics, such models closely resemble Ising spin models, extensively used in the study of crystal structure. In this note we are primarily interested in the computational properties of connectionist networks and their applicability to a range of computational problems. The information theoretical properties are discussed in [AM], [M].

3. The Hopfield model

A neural network with n nodes is modeled by an undirected simple graph $G = (V, E)$ on n vertices i.e. $|V| = n$. (Since the nodes are formally represented by the vertices of the graph, we shall use these two words interchangeably whenever the occasion demands). Each edge $e \in E$ has a real-valued weight $w_e$ associated with it. The edges represent the interconnections between the various nodes. The weight $w_{ij}$ of edge $(i, j) \in E$ (which could be negative) represents the strength of the synaptic connection between the two nodes i and j. Note that if nodes i and j are not connected in G then the weight between them can be assumed to be identically zero. Also note that since G is a simple undirected graph, $w_{ij} = w_{ji}$ and $w_{ii} = 0$, for all $i, j \in V$. The $j^{th}$ node can be in one of two states: $x_i = +1$ (on) or $x_i = -1$ (off). An example of the model is shown in Figure 1.

![Figure 1](image)

*Figure 1*

An example of the model

3.1. Statement of the Problem

Let $W$ represent the $n \times n$ weighted adjacency matrix of the graph $G$, i.e. $w_{ij}$ is the $(i, j)^{th}$ entry of matrix $W$. Let $x$ represent the $n \times 1$ state vector, whose $i^{th}$ entry is $x_i$, the state of vertex i.
Let the quadratic cost function $J(.)$ be defined by:

$$J(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_i x_j = \mathbf{x}^T \mathbf{W} \mathbf{x}$$

where $\mathbf{x}^T$ is the transpose of $\mathbf{x}$.

Now consider the two following problems:

1. **Local Minimum**

   **Input:** $\mathbf{W}$: the weight matrix of $\mathbf{G}$.
   
   **Output:** $\mathbf{x}^* \in \{+1, -1\}^n$ such that,
   
   $$\forall \mathbf{x} \in \mathcal{N}(\mathbf{x}^*), \quad J(\mathbf{x}^*) \leq J(\mathbf{x})$$
   
   where $\mathcal{N}(\mathbf{x}) \subset \{+1, -1\}^n$ is the set of $n$ vectors in $\{+1, -1\}^n$ which differ from $\mathbf{x}^*$ in exactly one component i.e. $\mathcal{N}(\mathbf{x})$ is the set of $n$ vectors at unit Hamming distance from $\mathbf{x}^*$.

   An alternative formulation is:

   **Output:** $\mathbf{x}^* \in \{+1, -1\}^n$ such that,

   $$\forall i \in \{1 \ldots n\}, \quad x_i^* \left( \sum_{j=1}^{n} w_{ij} x_j^* \right) \leq 0$$

   where $x_k^*$ is the $k$th component of the vector $\mathbf{x}^*$.

2. **Global Minimum**

   **Input:** $\mathbf{W}$: the weight matrix of $\mathbf{G}$.
   
   **Output:** $\mathbf{x}^* \in \{+1, -1\}^n$ such that,

   $$\forall \mathbf{x} \in \{+1, -1\}^n, \quad J(\mathbf{x}^*) \leq J(\mathbf{x})$$

   It is obvious that for each one of the above problems a solution exists. In the next sections we address the applications of local/global minima in connectionist networks.

3.2. **Local Minimum**

   It has been mentioned in the introduction that in the Hopfield model a local minimum configuration corresponds to an information storage point of the neural network. These storage points are stable, because if for some reason (like external disturbances) only one node changes its state, the system will, in most cases, go back to its original state. Furthermore, this tendency of the Hopfield model can be used to explain the phenomenon of content-addressable memory found in neural organizations.

   There are many computational problems, particularly in the area of simple undirected graphs, that can be cast into the framework of the Local Minimum problem. This is achieved by constructing a suitable weight matrix $\mathbf{W}$. One such problem is that of the Maximal Independent Set (MIS). An MIS of an undirected graph is a maximal collection of vertices $I$ subject to the restriction that no pair of vertices in $I$ are adjacent in the graph. Luby [Lu] shows that the problem of finding an MIS in a graph is a
1. Introduction

Though research in Artificial Intelligence (AI) has led to several important developments the fundamental problems in AI remain open. The theoretical studies indicate the many of the basic problems in AI are inherently intractable (NP-complete) if not worse (undecidable). At the same time humans seem to solve these problems quite accurately in the majority of cases in just a few milliseconds (vision, speech, natural language, planning, etc). During the recent few years a growing new class of algorithms has been proposed for Artificial Intelligence applications. These algorithms, sometimes called connectionist algorithms are implemented on a massively parallel network (MPN) that performs constrained optimization-like procedure. The algorithms in this class have several interesting characteristics:

1. The problem is formulated as an "energy minimization" process due to the analogy to well known physical processes in statistical mechanics [Ho],[Fe].

2. The algorithms can be implemented on large massively parallel networks of interconnected processors operating asynchronously [FHS].

3. The algorithms make use of few powerful constraint optimization techniques without resorting to a careful design that varies dramatically depending on the problem (as it has been traditionally done in theoretical computer science).

The above properties make this approach particularly interesting from the computational complexity perspective. The main advantage of the connectionist network formulation is its general purpose applicability to a broad range of problem domains [HT],[BH],[RM]. Thus, the method can be applied to new and unknown applications for which no previous algorithms are known. In this sense, the approach is analogous to such powerful constraint satisfaction procedures as linear programming, resolution based theorem proving and relaxation. Unlike these three methods, using the learning algorithms that have been recently developed the network can adapt to changing environments and improve its performance with time [RHW],[HSA].

A major problem that has not been addressed so far is the computational complexity of connectionist network algorithms. This question has two main aspects.

1. Determining the class of computations performed particularly well by these networks and the class of computations that are inherently unsuitable for connectionist implementation. Intuitively, given a problem $P$ of input size $N$, we would like to estimate whether a network of size $S(N)$ can solve the problem in time $T(N)$.

2. Determining the computational complexity of the optimization process performed by the network. Specifically, given a description of the network of size $N$, can we find subexponential algorithms to find stable (and maximally stable) states in the network that correspond to local and global energy minima (see Section 2 for details).

Time and space (memory) are the fundamental resources in the study of computational complexity of sequential machines. Thus, the first question to be answered is what is the analog of these in the context of connectionist networks. Naturally, time and network size are the two immediate candidates. However, we should also consider the number of states allowed at each of the nodes and the size of the arc-weights that contribute to the complexity of building the networks. Additionally, we may need to account for the number of layers (depth) of nodes in the network and the number of
special case of finding a local minimum in the Hopfield model. Another natural problem which is a special case of the Local Minimum problem is the Different Than Majority Labeling (DTML) problem. Given an undirected graph, the DTML problem is to label each vertex \( v \) with a label of +1 or -1 such that, at least half the neighbors of \( v \) have the label opposite to that of \( v \).

3.3. Global Minimum

In most optimization problems (see for example [Le]), it is the question of finding a global optimum which is of primary importance. In many of such problems, the best we can effectively achieve (without exhaustive search) is a local optimum. In several important cases, (as in linear programming or quadratic programming with a positive-definite matrix) a local optimum is also a global one. This fact may considerably simplify the search strategy. However this is not true in general.

In the Hopfield model a local minimum is all that the solution demands. The most stable state of the system, however is one corresponding to a global minimum. The Global Minimum problem is therefore very important in this context.

Moreover, even in problems where only a local optimum is sought, various researchers have commented on the quality of the local optimum found. (See for example [LK].) The quality of a local optimum can be judged by its closeness to a global optimum. Hence the importance of studying the Global Minimum problem.

The global problem is also important from a practical standpoint. Although the problem may be intractable, a good approximation algorithm / heuristic for it may lead us to or close to a local minimum.

Furthermore, in the context of constraint satisfaction networks, an approximate solution to a global minimum may be considered to be a better state than a high and shallow local minimum.

4. The Sequential (Asynchronous) Algorithm

In the introduction it was stated that every node possesses threshold-type computational capabilities. In this section, we explore the nature of these abilities and present an algorithm for solving the Local Minimum problem.

4.1. The Threshold Law

The dynamic behavior of a connectionist network consists of two parts:

1. The scheduling problem (updating Rate).
2. The transition rule

The scheduling problem relates to the decision of considering a node (a set of nodes) for a possible update. For example in the Hopfield model, the nodes update asynchronously and at random. In Nettalk, the nodes are updated sequentially in an ordered fashion. In [HSA] the nodes are updated according to a statistical Boltzmann distribution.

The transition rule describes the conditions for a node to change its current state.
The dynamic behavior of a connectionist network can be explained through the concept of firing, that is changing state when proper conditions are met. We define the behavior of the system as follows:

Let $x_j^-$ represent the state of node $j$ before $i$ is fired. Let $x_i^+$ be the state of the $i^{th}$ node after firing. Then,

$$x_i^+ = \begin{cases} 
-1 & \text{if } \sum_{j=1}^{n} w_{ij} x_j^- > 0 \\
+1 & \text{if } \sum_{j=1}^{n} w_{ij} x_j^- \leq 0
\end{cases}$$

That is, a node $x^*$ is stable if $x_i^* \left( \sum_{j=1}^{n} w_{ij} x_j^* \right) \leq 0$ (that is the local energy state is zero or negative). Note that in many of the papers on connectionist networks stability is defined in terms of a positive local energy state but this does not affect the results of the complexity analysis described here.

This Threshold Law provides the basis of the following sequential algorithm for finding a local optimum.

### 4.2. Algorithm 1

**Input:** $W$: the weight matrix of $G$

**Output:** A local optimum $x^* \in \{+1, -1\}^n$.

**Method:** Iteration is performed until the quadratic cost function $J$ does not change. Each iteration is a lexicographic search over all the vertices, wherein the state of every vertex is changed if necessary, in accordance with the Threshold Law.

The formal algorithm is presented in *Figure 2*.

It is well known that the following proposition is true. (The proof is straightforward and is omitted.)

**Proposition 1:** Algorithm 1 terminates in a finite number of steps and is correct.

In the next section we shall show that if the nodes are allowed to fire in parallel then convergence is not guaranteed. In fact such a scheme can lead to oscillations in the system.

### 5. The Parallel (Synchronous) Algorithm

Consider the two-vertex example shown in *Figure 3*. Here vertices 1 and 2 are joined by an edge of weight +1. The initial state of the system is $(+1, +1)$. Let each vertex update itself, synchronously, in accordance with the Threshold law. After a simultaneous update, the system is in state $(-1, -1)$. Another synchronous update now takes the system back to its original state $(+1, +1)$. Hence a cycle results.

Thus the synchronous parallel updating algorithm may not converge. In fact, it has been shown that (see [PM],[LM]), in general, a cycle of periodicity at most two can result under such a scheme.
begin
Choose any vector $x \in \{+1, -1\}^n$
Compute $J_{\text{new}} \leftarrow x^T W x$
repeat
\[ J_{\text{old}} \leftarrow J_{\text{new}} \]
for $i := 1$ to $n$ do
\begin{align*}
x_i &\leftarrow \begin{cases} 
-1 & \text{if } \sum_{j=1}^{n} w_{ij} x_j > 0 \\
+1 & \text{if } \sum_{j=1}^{n} w_{ij} x_j \leq 0 
\end{cases} 
\end{align*}
end
Compute $J_{\text{new}} \leftarrow x^T W x$
until ($J_{\text{new}} = J_{\text{old}}$)
end
The local minimum $x^* = x$

Figure 2
The Sequential Algorithm
Algorithm 1

6. Complexity Results

In the previous sections, the edge-weights in the Hopfield model were real numbers. However, in order to describe the complexity results in this section we make the assumption that the weights are chosen from the field of integers (positive and negative).

Since one of our primary interests lies in finding fast algorithms for the Local and Global Minimum problems, we shall look for algorithms whose worst-case running time is polynomially bounded in the number of vertices of the graph. Since this is impossible to achieve unless the input to the algorithms (which is the weight matrix $W$ in both cases) are themselves bounded by lengths (i.e. number of bits necessary to represent to represent the input) polynomial in the number of vertices $n$, we make the assumption that each of the $n^2$ entries in $W$ is a number requiring a polynomial number of bits for its binary representation. However even if the largest weight requires $n$ bits, it is theoretically plausible that Algorithm 1 will require $O(2^n)$ time to converge. In fact, in the case of directed networks we can construct a polynomial size circuit that simulates a counter and achieves this bound. This occurs because the actual weight of the edge can be of $O(2^n)$. An useful function in this context is the MAX function (see Garey and Johnson [GJ]) which can be defined in our case as follows:

\[ \text{MAX}(W) = \max_{(i, j) \in E} |w_{ij}| \]
Figure 3

Example demonstrating cycling in synchronous updates

Note that even if every edge-weight in the graph G requires at most n bits for its binary representation, \( \text{MAX}(W) \) can be exponential in n.

However,

**Proposition 2:** If \( \text{MAX}(W) \) is polynomially bounded in n, then Algorithm 1 takes a polynomial number of steps to converge to a local minimum.

The proof is straightforward and is omitted.

Since it is possible for \( \text{MAX}(W) \) to assume values which are exponential in n, we would like to find out whether there exists a \( n \times n \) weight matrix \( W \) and an initial starting vector \( x \in \{+1, -1\}^n \) for which Algorithm 1 requires exponential time to converge to a local minimum. This is the first major open problem that we would like to address because the majority of current algorithms for finding local optima are but variations of Algorithm 1. This is in analogy with the Simplex Method which behaves very well on the average, but has exponential worst-case time complexity.

In the case of the Global Minimum problem, the quest for a fast algorithm becomes provably harder. Consider the decision problem corresponding to the Global Minimum.

**Global Minimum Decision (GMD) Problem**

*Instance:* \( W \): the weight matrix of G and integer \( K < 0 \).

*Question:* Does there exist a \( n \times 1 \) vector \( x \in \{+1, -1\}^n \) such that,

\[
J(x) = x^T W x \leq K
\]
Proposition 3: The GMD problem is strongly NP-complete.
The proof is by transformation from MAX CUT (see Karp [Ka]).
This leads to the following:

Proposition 4: The Global Minimum problem is NP-hard.

7. Graph-Theoretic Reformulation

Consider any $n \times 1$ vector $x$ in $\{+1, -1\}^n$. Note that since

$$J(x) = x^t W x = (-x)^t W (-x) = J(-x)$$

both the Local and Global Minimum problems remain invariant under sign interchanges in the states. Thus the problem is unchanged if all states with label $+1$ were changed to a label of $-1$ and vice versa. Consequently, the problem can be viewed as a problem of partitioning the vertex set $V$ of the graph $G$ into two sets $V_1$ and $V_2$ ($V_1 \cup V_2 = V$, $V_1 \cap V_2 = \emptyset$), vertices of the same label being in the same subset. Without loss of generality, we shall henceforth follow the convention that all vertices with state $+1$ are in $V_1$ and all vertices with state $-1$ are in $V_2$. Note that there is a bijection between every vector $x \in \{+1, -1\}^n$ and every bipartition $(V_1, V_2)$ of the vertex set $V$ of the graph $G = (V, E)$.

Consider any vector $x \in \{+1, -1\}^n$. The quadratic cost function $J(x)$ associated with this vector can be rewritten in terms of the bipartition $(V_1, V_2)$ corresponding to $x$ as:

$$J(x) = 2T - 4 \sum_{u \in V_1, v \in V_2} w_{uv}$$

where, $T = \sum_{e \in E} w_e$.

Any minimization of the cost function $J$ thus entails a corresponding maximization of the interpartition sum i.e. the sum of the weights over all the edges across the bipartition. Formally, if $H = (V, E_H)$ is the spanning bipartite subgraph of $G$ corresponding to the bipartition $(V_1, V_2)$ (where, $E_H = \{(u, v) \in E \mid u \in V_1 \& v \in V_2\}$) then the interpartition sum is given by $\sum_{e \in E_H} w_e$. We can now reformulate our Local and Global Minimum problems as follows.

7.1. Local Minimum

Informally, this problem seeks a bipartition $(V_1^*, V_2^*)$ of the vertex set $V$ such that, if we move any one vertex $v$ from one subset to the other i.e. either from $V_1^*$ to $V_2^*$ or from $V_2^*$ to $V_1^*$, the interpartition sum decreases. It can easily be shown that this condition is equivalent to the following reformulation of the Local Minimum problem.

1. Local Minimum

Input: The edge-weighted graph $G$.

Output: A spanning bipartite subgraph $H$ of $G$ such that,

$$\forall v \in V, \quad \delta_H(v) \geq \frac{1}{2} \delta_G(v)$$
where, for any edge-weighted simple undirected graph \( A = (V_A, E_A) \), the weighted degree \( \delta_A(v) \) is defined for all \( v \in V_A \) as \( \sum_{(u, v) \in E_A} w_{uv} \).

7.2. Global Minimum

It has been observed that a minimization of the quadratic cost function \( J \) corresponds to a maximization of the interpartition sum. Hence a global minimum of \( J \) corresponds to a global maximum of the interpartition sum. Consequently, the Global Minimum problem can be reformulated as follows.

2. Global Minimum

\[ \text{Input:} \quad \text{The edge-weighted graph } G. \]

\[ \text{Output:} \quad \text{A spanning bipartite subgraph } H \text{ of } G \text{ such that,} \]

For all spanning bipartite subgraphs \( H' \) of \( G \), \( \sum_{e \in E_H} w_e \geq \sum_{e \in E_{H'}} w_e \),

or in other words a Maximum Cut.

The MAX CUT problem (see Karp [Ka]) is a restricted version of this problem, where, all the weights are chosen from the set of non-negative integers.

7.3. Feasible Partitions

From the condition for a Local Minimum shown above, it should be clear that a necessary condition for \( H \) to correspond to a local minimum is \( \sum_{e \in E_H} w_e \geq \frac{1}{2} \sum_{e \in E_G} w_e \).

Any partition of the vertex set \( V \) which satisfies the above condition is called a Feasible Partition. Obviously the set of local minima is a subset of the set of Feasible Partitions.

8. The Approximation Algorithm

In this section we shall present a method of reaching a Feasible Partition in polynomial time. Indeed it is a \( \frac{1}{2} \) approximation algorithm to the MAX CUT problem.

From the viewpoint of finding a local minimum, this algorithm is important because it would provide in polynomial time a good starting vector to run Algorithm 1 on.

8.1. Algorithm 2

\[ \text{Input:} \quad W: \text{the weight matrix of } G \]

\[ \text{Output:} \quad \text{A Feasible Partition } (V_1^*, V_2^*) \text{ of the vertex set } V. \]

\[ \text{Method:} \quad \text{The method consists of creating a state vector } x \text{ for a Feasible Partition. The set of vertices having the the state label } +1 \text{ then goes into vertex set } V_1^*, \text{ and the rest having state label } -1 \text{ go into } V_2^*. \text{ The algorithm considers only the superdiagonal entries of the matrix } W \text{ and does a causal updating on the state of each vertex, from } n \text{ to } 1. \]

The formal algorithm is presented in Figure 4.
begin
    \( x_i \leftarrow 1 \)
    for \( i := (n - 1) \) to 1 step -1 do
        begin
            \[
x_i \leftarrow \begin{cases}
                -1 & \text{if } \sum_{j=i+1}^{n} w_{ij} x_j > 0 \\
                +1 & \text{if } \sum_{j=i+1}^{n} w_{ij} x_j \leq 0
            \end{cases}
            \]
            (This constitutes a causal update)
        end
        Choose \( V_1^* = \{ i \mid x_i = +1 \} \) & \( V_2^* = V - V_1^* \).
    end
    \((V_1^*, V_2^*)\) is a Feasible Partition of vertex set \( V \).
\[\text{Figure 4}\]
\text{The Approximation Algorithm}
\text{Algorithm 2}

The following propositions concerning the algorithm are now stated without proof.

**Proposition 5:** Algorithm 2 is a polynomial time algorithm.

**Proposition 6:** Algorithm 2 is a \( \frac{1}{2} \) approximation algorithm to the MAX CUT problem.

Proposition 6 means that if \( H^* \) is the spanning bipartite subgraph corresponding to a global minimum, and \( H \) is the spanning bipartite subgraph corresponding to the bipartition obtained by Algorithm 2, then,

\[
\sum_{e \in E_{H^*}} w_e \leq 2 \sum_{e \in E_H} w_e
\]

where, all the weights \( w_e \) are non-negative integers.

Several other ideas can be used to provably improve the performance of this algorithm. However they will require a substantially lengthier discussion and consequently will not be mentioned here.

We shall however mention in this context an interesting, though somewhat disheartening intractability result. Consider the following problem:

**Relative Sign (RS) Problem**

**Instance:** \( W \): the weight matrix of \( G \) and two specified vertices \( s, t \in V \).
Question: Does there exist a local minimum in which the vertices \( s \) and \( t \) receive opposite signs.

Note that if it were possible to receive an answer to this question in polynomial time, then we could easily find a polynomial time solution to the Local Minimum Problem, by asking this question to all possible unordered 2-tuples of vertices. However we can show that:

**Proposition 7:** The RS problem is NP-complete.

The proof follows from a reduction to the PARTITION problem (see Karp [Ka]). Consider the edge-weighted graph presented in Figure 5. It can easily be shown that the specified vertices \( s \) and \( t \) have opposite signs if and only if there exists a partition of the weights \( w_1, w_2, \ldots, w_n \).

![Figure 5](image)

**Figure 5**
Graph used in transforming PARTITION to RS

9. Heuristics

There are several heuristics aimed at speeding up the process of finding a local minimum that we have investigated. The difference between them lies in the fact that the first heuristic (like Algorithm 1) finds a local minimum, the second heuristic (like Algorithm 2) provides a good estimate of the local minimum. We shall give an informal description of these heuristics below.
9.1. Heuristic 1

**Input:** \( W \): the weight matrix of \( G \)

**Output:** A local optimum \( x^* \in \{+1, -1\}^n \)

**Method:**

1. Start at any vector \( x \) in \( \{+1, -1\}^n \).

2. Check if any vector at Hamming distance 1 from the current vector has a lower cost function \( J \) associated with it.

3. If exactly one such vector is found (in Step 2) make that the current vector.
   
   If more than one such vector is found (in Step 2) make the one which has the lowest cost function associated with it the current vector. Break ties lexicographically.
   
   Go back to Step 2.

4. If no such vector is found (in Step 2) then stop. The current vector is a local minimum.

The same heuristic can be reformulated with a graph-theoretic flavor (from the discussion in Section 6). For the sake of completeness, this is provided below.

**Input:** The edge-weighted graph \( G \).

**Output:** A partition \( (V_1^*, V_2^*) \) of the vertex set \( V \) of \( G \), corresponding to a local minimum.

**Method:**

1. Start with any partition \( (V_1, V_2) \) of the vertex set \( V \). Let, \( H \) be the corresponding spanning bipartite subgraph of \( G \).

2. For every vertex \( v \) in \( V \), check if \( \delta_G(v) - 2\delta_H(v) \leq 0 \), where \( H \) is the spanning bipartite subgraph of \( G \) corresponding to the current partition.

3. If the condition in Step 2 is not satisfied then consider the vertex \( u \) for which this function is a maximum (over all the vertices in \( V \)). Break ties lexicographically.
   
   Move \( u \) to the other subset i.e. if \( u \) is in \( V_1 \) move \( u \) to \( V_2 \) and vice versa.
   
   Make this configuration the current bipartition of \( V \).
   
   Go back to Step 2.

4. If the condition in Step 2 is satisfied, then, the current bipartition corresponds to a local minimum.

It is obvious from the definition of a local minimum, that such an algorithm would work. It is probably better than Algorithm 1 because it can take *deeper stabs* in the search space. (See for example Papadimitriou and Steiglitz [PS] for local search algorithms of this kind.)

It can be shown that finding an optimal sequence of moves to a local minimum is itself an NP-hard problem.

We now consider the second heuristic, which should give us a good estimate for a local minimum.
9.2. Heuristic 2

**Input:** The edge-weighted graph G.

**Output:** A good estimate of a local minimum.

**Method:**

1. **Step 1.** Find a maximum spanning tree in the graph G by Kruskal’s or Prim’s algorithm.

2. **Step 2.** A tree is a bipartite graph. Find the bipartition. (This can be done quite easily in linear time.)

3. **Step 3.** Take this bipartition of the vertex set V as the estimate of the local minimum.

This heuristic ought to work quite well because very many heavily weighted edges will now appear in the spanning bipartite subgraph H and (from the discussion in Section 6) will contribute greatly in reducing the cost function J.

10. Summary

In this section we summarize the main open problems. In this note we focused on the computational complexity of connectionist networks with symmetric connections ($w_{ij} = w_{ji}$). The main reason for this is the fact that directed (asymmetric) networks can be shown to simulate boolean combinatorial circuits, which have been extensively studied in computer science.

As mentioned in the introduction there is an important distinction in the complexity studies of connectionist networks between two related but possibly very different questions.

(*) The range of computable functions on a connectionist network. More precisely given a problem of input size N can we construct a connectionist network of size $S(N)$ that can solve the problem in time $T(N)$. This is in direct analogy to computational complexity of Turing machine computations as given in terms of tape length and computational time.

(**) The computational complexity of problems posed in terms of energy minimization on a connectionist network. That is, given a network comprised of N-nodes, and assuming maximal weight size of $\text{MAX}(W)$, what is the computational complexity of achieving local/global minima.

In the previous section we addressed several problems that belong the latter category. The problems we discussed were:

1. Can local optima be found sequentially in polynomial time?
2. Does the currently predominant method of obtaining local optima converge in polynomial time?

Obviously, a affirmative answer to (2) implies a affirmative answer to (1) but the converse does not necessarily hold. These investigations are critical since they will shed light on the question whether energy minimization on a connectionist network can be done in polynomial time (independent of a specific algorithm employed to solve the problem).
We have performed computer simulations on graphs with randomly chosen edge-weights. The results seem to indicate that Heuristic 1 converges to a local minimum in linear time.

We also addressed the problem of obtaining efficient approximation algorithms for global minima. There are several other important problems that we are planning to address in our investigations.

1. What is the parallel complexity of obtaining local optima. This question has two important aspects that relate to categories (*)) and (**)) respectively:
   (a) Can we devise highly parallel updating procedures that will be guaranteed to converge to a local minimum.
   (b) Can we devise a parallel algorithm (not necessarily a network algorithm) to find local minima.

As mentioned before [Lu] devised and efficient parallel algorithm for a special case of the connectionist network: the maximal independent set problem.

Once the fundamental questions above are resolved there are many interesting questions which we are planning to address. The majority of these questions address the range of computations admitted by networks with given structural (graph theoretical) properties. Of special interest are planar, bounded degree and positive weight networks. For instance, we already discovered a polynomial time algorithm for the problem of finding local minima in tree-like networks.

To summarize, we believe that a systematic theoretical treatment of the computational aspects of massively parallel networks is a necessary prerequisite for their acceptance as a feasible computational mechanism. Progress in this area will advance our knowledge of highly parallel machines and their applicability to perform complex tasks.

11. References


