Bitstream Neurons for Graph Colouring

Peter Burge
John Shawe-Taylor
Royal Holloway University of London

We introduce the Multi State Bitstream Neuron. By replacing the stochastic activation function with stochastic weights the MSBSN is shown to approximate a Generalised Boltzmann Machine. Benchmarks show the algorithm performs as well as the Boltzmann algorithm whilst the MSBSN lends itself to a very compact and fast hardware implementation.

1 INTRODUCTION

In [1] Shawe-Taylor and Žerovnik introduced the Generalised Boltzmann Machine (GBM) as an extension to the Boltzmann machine that enables us to map constraint problems, requiring more than two states, onto a recurrent neural network. In [2] experiments were performed using the Mean Field Annealing approach to graph colouring using the Petford and Welsh algorithm [3] as a GBM.

In this paper, we extend the use of bitstreams from the bi-polar, stochastically connected Boltzmann machine [4] to a recurrent network of stochastically connected multi-state bit stream neurons (MSBSN). We compare the performance of the resultant network with the results obtained using Mean Field Annealing described in [2] and the Petford and Welsh algorithm [3].

The results obtained from simulations of the MSBSN show that there is no increase in the number of iterations taken to solve a typical randomly generated k-colourable graph. Clearly the digital nature of the neuron combined with its simple functionality, described by Shawe-Taylor, Van Daalen and Zhao in [5], could lead to a fast and inexpensive hardware implementaion.

[0] The authors would like to thank M. Van Daalen and J. Žerovnik for helpful discussions.
Correspondence and requests for reprints should be sent to Peter Burge and John Shawe-Taylor, Department of Computer Science, Royal Holloway, University of London, Egham, Surrey, England. TW20 OEX. Email Peteb@dcs.rhbnc.ac.uk, john@dcs.rhbnc.ac.uk .

The results also demonstrate that the standard method of introducing stochasticity in a Boltzmann Machine, namely the stochastic update rule, can be replaced by a stochastic connectivity between the multi-state neurons. It is this approach that opens the way for efficient hardware implementation.

2 THE GENERALISED BOLTZMANN MACHINE

The Generalised Boltzmann machine introduced by Shawe-Taylor and Žerovnik [1] is a natural progression from the Boltzmann Machine into a solution space of multiple states.

The GBM is defined, in [2], as follows;

Let A be a finite alphabet of r symbols and $G = (N, E)$ be any graph with vertex set N and edge set E. A Generalised Boltzmann Machine $\beta(G, A)$ on G over A is specified by a mapping ω from the set E into the set of matrices with real entries indexed by $A \times A$ and a mapping z from the set N to vectors indexed by A. The matrices will be referred to as weights and the vectors as thresholds. A state of the machine is an assignment σ which specifies for each node a symbol from A. The state of a node v is its value under the assignment σ . We define the Energy of a state σ to be the quantity

$$\xi = \xi(\beta, \sigma) = -0.5 \sum_{(u,v) \in E} \omega(u, v)_{\sigma(u)\sigma(v)} + \sum_{u \in N} z(u)_{\sigma(u)}$$

When considering the graph colouring problem, we let $z(u)_{\sigma(v)} = 0$. There is an individual weight matrix of size $r \times r$ for each adjacent edge. For the graph colouring problem we simply set each of these matrices to be the negative identity matrix. This clearly results in an increase in overall energy when two adjacent nodes share the same colour i.e.when $\sigma(u) = \sigma(v)$.

3 STOCHASTIC BITSTREAMS AND THE MULTI-STATE BITSTREAM NEURON

A stochastic bitstream is a vector of binary values where the frequency of 'ones' is directly proportional to a real value that the bitstream represents. The multi-state bitstream neuron is a digital processing unit capable of performing very low level operations on these stochastic bitstreams. The k-state bitstream neuron has k counters and each counter has a weighted connection to each of its input neurons $x^1, ..., x^n$. Let $W^\ell, \ell = 1, \ldots, n$ denote the matrix of weight bitstreams from input ℓ to the counters. Hence, W^ℓ_{ij} is the weight bitstream from state i of input x^ℓ to counter j. At time t the neuron is considered to be in one of the k states, denoted by $S(t), t = 1, 2, ...$ The weights have values in the range [-1,1] and are represented by appropriate bitstreams and sign bits. During one operational cycle at time t

the neuron takes one bit w_{ij}^{ℓ} from each weight bitstream W_{ij}^{ℓ}. The neuron then computes the values

$$C_j = \sum_{\ell=1}^{n} w^{\ell}_{\sigma(\ell)j} \mathrm{sgn}(W^{\ell}_{\sigma(\ell)j}), \quad (1)$$

where $\sigma(\ell)$ is the state of neuron x^{ℓ} at time t. The state of the neuron is subsequently updated to

$$S(t+1) = \mathrm{argmax}_j\{C_j\}. \quad (2)$$

In the case of ties the choice is resolved by throwing an appropriately sided dice.

4 THE MSBSN ALGORITHM APPLIED TO GRAPH COLOURING

The graph colouring problem maps directly onto a Generalised Boltzmann machine by setting the weight matrix associated with a graph edge to be the negative identity matrix. Hence, weights are either -1 or 0. To simplify the computation we swap the signs of the weights and compute the new state using *argmin* in place of *argmax*. The key to the stochastic update rule required for the Boltzmann style search is to use the bitstream weights. This is achieved by setting the non zero weights equal to a pseudo temperature parameter $b \in [0, 1]$. When $b = 0$ the updates are completely random corresponding to a Boltzmann temperature of $T = \infty$. As b increases to $+1$, the operation becomes a localized gradient descent corresponding to $T = 0$. The GBM would use fixed weights ($b = 1$) and make a stochastic state allocation based on the counters values. Hence, the stochasticity of the 'activation' function has been replaced by stochastic weights.

In our experiments, the network was run at a constant intermediate temperature. In the majority of cases, the temperature proved to be high enough to avoid the network getting trapped in local minima. The algorithm follows:

Initialise bitstream weights to a value $b \in [0, 1]$.
While (graph not coloured correctly) {
    Randomly pick a node to update;
    Loop (over all adjacent nodes){
        Pick current entry from associated weight bitstream;
        If this entry is a one {
            increment counter which is of the same value as the state
            of the adjacent node under consideration.}}
    Set the state of the current node to be that of the counter with the lowest value.
    If there is more than one counter with the same lowest value,
    randomly pick one from this group. }

5 EXPERIMENTAL RESULTS

To produce a fair comparison among the algorithms,the graphs to be coloured were randomly generated prior to the experiments. This meant all the algorithms would be tested on the same data. Each graph was created with n nodes ranging from $n = 50$ to $n = 400$. The nodes were then divided into three equal groups. The probability of an edge connection between two nodes in different groups was $p = 0.5$, 0.1 and 0.05. In each case the graphs were designed to be three colourable by forbidding connections within each of the groups. For each experimental configuration, one hundred repetitions of the experiment were performed and an average of the results taken. From earlier trials, we chose a cut-off level of 9000 itterations to be the point at which a configuration was considered unsolved. By taking an average over all 100 repetitions, including those that had reached the cut-off level, we acheaved a statistical value that incorporated both ability to reach a solution and speed of convergence. For MFA the temperature was set to the implied temperature of the Petford & Welsh algorithm. Figures one to three are included to show that the MS-BSN algorithm performs equally as well as MFA and the Petford & Welsh algorithm.

FIGURE 1, ALGORITHM PERFORMANCE COMPARISON, C=3, P=0.5

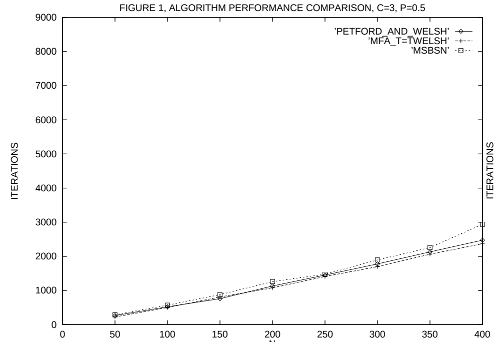


FIGURE 2, ALGORITHM PERFORMANCE COMPARISON, C=3, P=0.1

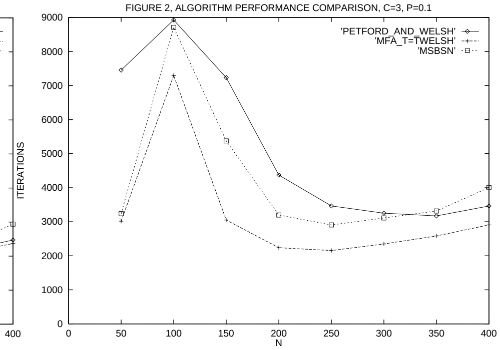
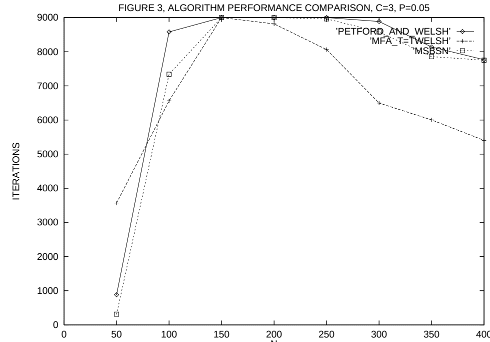


FIGURE 3, ALGORITHM PERFORMANCE COMPARISON, C=3, P=0.05



6 CONCLUSION AND FUTURE WORK

The results show that a recurrent network of MSBSN's is capable of solving optimisation problems, in particular Graph Colouring. There is no adverse drop in performance when using the MSBSN algorithm compared with MFA and the Petford & Welsh algorithm. The MSBSN lends itself to a hardware implementation with simple logical functionality. It therefore, has the potential to be implemented for large systems at very fast speeds. For a system involving 400 nodes running on a 20MHz hardware implementation we estimate a speed up factor of 200 over running the algorithms on a Sparc 5 model 85.

This paper has demonstrated that the activation stochasticity of the standard Boltzmann model can be replaced by stochastic weights in the multi-state model. At the same time this approach opens the way to a very fast hardware implementation of the techniques with apparently no loss of solution quality.

References

[1] J.Shawe-Taylor, J.Žerovnik. Generalised Boltzmann Machines, Technical Report CSD-TR-92-29, Royal Holloway University of London, 1992.

[2] J.Shawe-Taylor, J.Žerovnik. Analysis of the Mean Field Annealing Algorithm for Graph Colouring. Internal Report, Royal Holloway University of London, 1993.

[3] A.D. Petford, D.J.A. Welsh. A randomised 3-colouring algorithm, Discrete Mathematics, Volume 74, pages 253-261, 1989.

[4] J.Zhao, J.Shawe-Taylor. Stochastic Connection Neural Networks, Proceedings of Fourth IEE Conference on Artificial Neural Networks, IEE Conference Publication 409, pages 35-39, 1995.

[5] J.Shawe-Taylor, M.Van Daalen, J.Zhao. Learning in Stochastic Bit Stream Neural Networks. To appear in Neural Networks.

[6] J.Shawe-Taylor, P.Jeavons & M.Van Daalen. Probabilistic Bit Stream Neural Chip: Theory. Connection Science, Volume 3, No. 3, 1991.