Adapting the Energy Landscape for MFA

Peter Burge
John Shawe-Taylor
Royal Holloway University of London

We combine Mean Field Annealing with an anti-hebbian type adaptive weight penalty method forming an algorithm that performs well on standard benchmark optimization problems. We compare the hybrid algorithm with the Petford and Welsh algorithm, MFA at a constant temperature and a stochastic weight penalty technique, known as GENET.

1 INTRODUCTION

Combinatorial optimization problems such as the Traveling Salesman Problem (TSP), Graph Bi-partitioning and Graph Colouring have been used extensively for bench marking new optimization algorithms. The challenge is to develop a general purpose algorithm that converges rapidly and yet still produces a high quality solution.

When an optimization problem is mapped onto a neural network as states of nodes connected by edges, we assign the problem a 'Cost' or 'Energy' function. This energy is a function of the global state of the network at any one instant and the energy function itself models the problem at hand. High energy represents a state that is far from the desired solution. As we iterate through a node updating procedure, we change the states of nodes and attempt to reduce the overall energy, eventually finding a near optimal solution. The solution space for the problem can be thought of as a multi-dimensional energy landscape containing deep valleys or 'local minima'. Local minima can trap a gradient descent technique resulting in a high energy solution being found.

Simulated Annealing (SA) [1] appears to provide us with the best compromise between rate of convergence and solution quality. Transitions to states that increase

0

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 overall energy are permitted in addition to energy decreasing updates. This flexibility to increase the overall energy allows the system to escape from local minima. The percentage likelihood of accepting a proposed state transition that will increase the overall energy is governed by a pseudo temperature parameter T. The rate of change of this temperature is governed by a 'Cooling Schedule'. It has been shown (Aarts & Korst [1]) that with a logarithmic cooling schedule, we can guarantee convergence to a global minimum with probability one, over a sufficiently long period of time. Unfortunately, in practice the time required makes the approach impractical for all but small problems.

In 1987 Peterson and Anderson introduced the Mean Field Learning Algorithm (MFLA) for Neural Networks [6]. They replace the stochastic update of SA with deterministic mean field theory equations. Again they use a pseudo temperature parameter with a cooling schedule and reported speed up factors of up to 30 over SA, with only a slight reduction in solution quality.

The GENET algorithm, (Tsang & Wang (1992) [10]), is a stochastic technique comprising two stages. Gradient descent has control initially until the system stabilizes in a minimum. If trapped in a non-zero energy state, a breakout method takes over. This breakout method, proposed by Morris in 1993 [5], uses a weight penalty matrix to penalise connections that are consistently adding to the overall energy. This localised penalty effectively changes the shape of the energy landscape around the local minimum raising the surface until the system can once again follow a gradient descent path.

We observe that Tsang's update rule is essentially anti-Hebbian. This suggests that we can apply the same technique to the MFA algorithm giving rise to a new algorithm in which the weights are continually updated hence avoiding the need to have two separate stages. The hybrid algorithm also results in an automatic cooling schedule where updating the weights effectively reduces the temperature.

We also include experiments comparing the performance of different algorithms on a series of controlled benchmark problems which allow us to distinguish difficult and easy examples. In the majority of cases, our hybrid algorithm outperforms both a stochastic algorithm and the MFA algorithm.

2 BENCHMARKING ADAPTIVE MFA

In this section we describe the Graph Colouring Problem which we used to benchmark the algorithms.

Definition 2.1 Given a number of colours C and a graph $G(V, E)$ where V is the set of vertices and E the set of pairs of vertices denoting the edge set, we aim to see if it is possible with C colours to colour each vertex of the graph such that all adjacent vertices are coloured differently.

This problem maps simply onto a recurrent neural network where the nodes represent the vertices of the graph colouring problem and the connections represent

the edges. Every node of the network has associated with it a state vector of length C, where C is the number of colours that we are attempting to colour the graph with. We denote by $\mathbf{V_i} = (V_{ij})_{j=1}^{C}$, the state vector for node i. Each state vector represents a probability distribution of colours for a node where V_{ij} is the probability of node i having colour j, hence, $\sum_{j=1}^{C} V_{ij} = 1$.

We will also refer to the state vector as the colour assignments. Each edge (i, k) of the network has an associated weight W_{ik} which together form a matrix. The input for colour i at a node consists of the product of the outputs from adjacent nodes for colour i with the corresponding weight from the weight matrix. The values are then then negated, exponentiated and normalised, giving an update rule for a randomly selected node i of

$$V'_{ij} = \frac{\exp(-\sum_{k\sim i} V_{kj} W_{ik})}{\sum_{l=1}^{C} \exp(-\sum_{k\sim i} V_{kl} W_{ik})} \qquad ; j = 1, \ldots, C \tag{1}$$

where $k \sim i$ means 'k adjacent to i'.

We adapt the Weight Matrix so that each non zero entry corresponding to a connection from node i to some other node k is incremented by a penalty value derived from the chord distance defined by Edwards and Cavilli-Sfroza [3]

$$\Delta W_{ik} = \langle \tilde{\mathbf{V}}_\mathbf{i}, \tilde{\mathbf{V}}_\mathbf{k} \rangle \tag{2}$$

where $\tilde{\mathbf{V}}_\mathbf{i} = \left(\sqrt{V_{ij}}\right)_{j=1}^{C}$. Note that the vector $\tilde{\mathbf{V}}_\mathbf{i}$ has unit length since $\langle \tilde{\mathbf{V}}_\mathbf{i}, \tilde{\mathbf{V}}_\mathbf{i} \rangle = \sum_{j=1}^{C} V_{ij} = 1$ so ΔW_{ik} is the cosine of the angle between the corresponding vectors. Hence, the algorithm is based on the anti-hebbian rule tending to increase the penalty associated with connections where there is a significant conflict between the colour assignments of the linked vertices.

Before running the algorithm, the entries in the Weight Matrix W_{ik}, corresponding to a connection between node i and node k, are set to a constant value. Other entries are initialised to zero. The state vectors on each neuron are initialised such that $V_{ij} = \frac{1}{C}$. A small random perturbation about this value is then introduced throughout each state vector.

Using the above equations, the algorithm is as follows;

Initialise the Weight Matrix and State Vectors.
While (graph not coloured correctly) {
Randomly pick a node i to update.
Update V_{ij} to V'_{ij} given by equation 1.
For all nodes k adjacent to i set $W_{ik} = W_{ik} + \Delta W_{ik}$ given by equation 2.}

3 EXPERIMENTAL RESULTS

To make a fair comparison of our algorithm with others, we chose only ones that were sequential in nature. Firstly, the Petford and Welsh algorithm [7] which amounts to simulated annealing at a constant temperature. Aarts and Korst [1] point out that annealing at a well chosen constant temperature is almost as good as applying a cooling schedule. Secondly, we chose MFA with the temperature set to the implied temperature of the Petford and Welsh algorithm denoted in the figures by 'TWELSH'. Finally we used a sequential adaptation of the GENET algorithm which doesn't require a temperature but uses a breakout method when no decreases in energy are produced, i.e. when the system gets trapped in a local minimum.

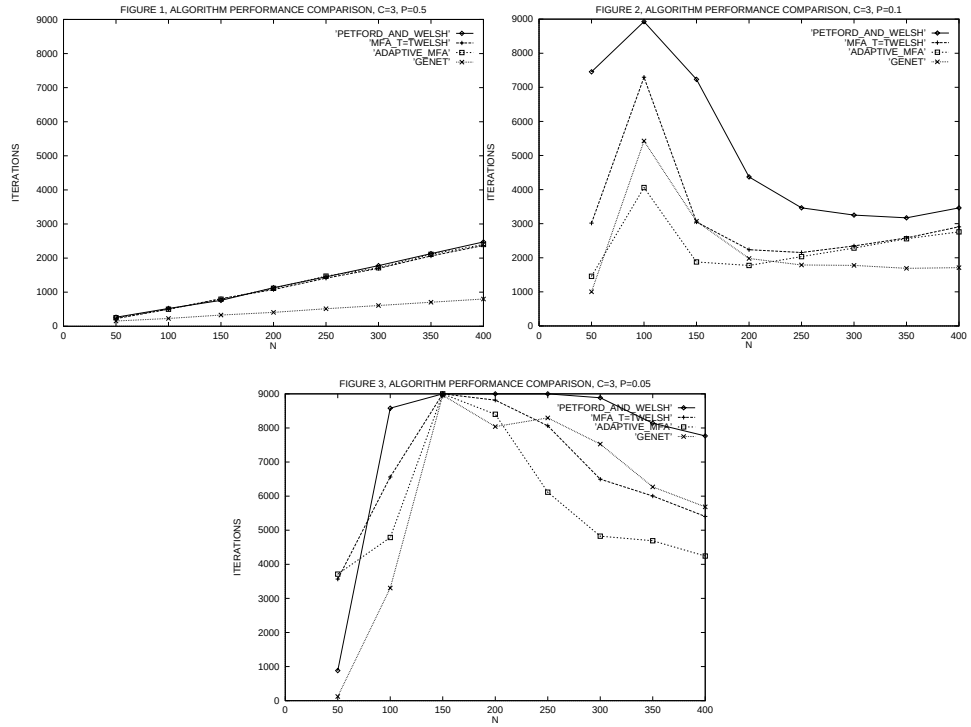
All four algorithms were tested on the same input data. We randomly generated three-colourable graphs by dividing the vertex set into three subsets of equal size. Three types of three-colourable graphs were generated by varying the probability p with which nodes in different subsets were made adjacent. The probability of two nodes being adjacent was $p = 0.5$, $p = 0.1$ and $p = 0.05$ respectively. The configurations $p = 0.1$ and $p = 0.05$ have interesting regions in which graphs prove difficult to colour [7]. This occurs when $n \approx \frac{8}{p}$. It was in this 'difficult' region that Adaptive MFA proved very successful.

For each value of p and n 100 graphs were generated. The graphs were generated probabilistically so that the same set of graphs were given to all the algorithms to ensure a fair trial. From early trials on the graphs, a cutoff level of 9000 iterations was picked as the point at which the algorithm was considered to have failed. By averaging over the full 100 trials for each configuration including those which had reached the cutoff level, we derived a general measure of performance comprising ability to come up with a solution and speed of convergence. For each connection density 100 repetitions were performed for $n = 50$ to $n = 400$ in increments of $\Delta n = 50$. Figures 1 to 3 show the results of our experiments.

Figure 1 is the case where $p = 0.5$ and so the connectivity is quite high. There is no identifiable region that proves difficult to solve. Adaptive MFA in this case performs equally well as the Petford and Welsh Algorithm and MFA. The GENET algorithm, however, is clearly better at solving graphs with high connectivity. This is because simple gradient descent is usually sufficient in this case, and the weight update stage was not needed.

In Figure 2, the connectivity has been reduced to $p = 0.1$. We identified a region around $n = 100$ where there was a marked increase in the average number of iterations required to reach a solution. In this 'difficult' region, our Adaptive MFA algorithm performs noticeably better than all the others.

Finally for the very sparsely connected graphs where p=0.05, we notice a widening of the 'difficult' region. From $n = 50$ to $n = 100$ GENET performs better than Adaptive MFA. From $n = 200$ on, we noticed that again Adaptive MFA comes out as the best algorithm.



In conclusion, we propose that the hybrid combination of MFA with an adaptive weight penalty technique is an algorithm worthy of further investigation. On the limited trials performed, Adaptive MFA seems to perform better on the whole in 'difficult' regions than the Petford and Welsh algorithm, MFA and GENET. The algorithm is simple, comprising of an update rule followed by a weight modification. Continual local annealing that adapts the energy landscape is implemented cleanly as an anti-hebbian weight change.

We propose further experiments investigating how we can control the weights at critical temperatures [9], where the critical paths bifurcate, by monitoring the sizes of updates.

References

[1] E.Aarts and J.Korst, Simulated Annealing and Boltzmann Machines. John Wiley & Sons. ISBN 0-471-92146-7.

[2] A.Davenport, E.Tsang, C.Wang and K.Zhu. GENET: A Connectionist Architecture for Solving Constraint Satisfaction Problems by Iterative Improvement. Department of Computer Science, University of Essex.

[3] A.Edwards, L.Cavalli-Sfroza, Phylogenetic analysis Models and estimation procedures, Amer. J. Hum. Genet. 19, 233-257, 1967.

[4] J.Hertz, A.Krogh, R.Palmer. Introduction to the theory of neural computation. Addison-Wesley Publishing Company. ISBN 0-201-51560-1.

[5] P. Morris. The breakout method for escaping from local minima. Proceedings of the Twelfth National Conference on Artificial Intelligence. AAAI Press/The MIT Press, 1993.

[6] C. Peterson, J. Anderson. A Mean Field Theory Learning Algorithm for Neural Networks, Complex Systems volume 1, pages 995-1019.

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

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

[9] J.Shawe-Taylor, J.Žerovnik. Analysis of the Mean Field Annealing Algorithm for Graph Colouring. Report CSD-TR-93-13, Royal Holloway University of London, 1993.

[10] E.Tsang, C.Wang. A Generic Neural Network Approach For Constraint Satisfaction Problems. In J.G.Taylor (ed.), Neural Network Applications, Springer-Verlag, pages 12-22, 1992.