Optimization with Neural Networks

Bo Söderberg

Department of Theoretical Physics, Lund University, Sölvegatan 14A, S-22362 Lund, Sweden

Abstract. The recurrent neural network approach to combinatorial optimization has during the last decade evolved into a competitive and versatile heuristic method, that can be used on a wide range of problem types.

In the state-of-the-art neural approach the discrete elementary decisions (not necessarily binary) are represented by continuous Potts mean-field neurons, interpolating between the available discrete states, with a dynamics based on iteration of a set of mean-field equations. Driven by annealing in an artificial temperature, they will converge into a candidate solution.

1 Introduction

Combinatorial optimization problems often require a more or less exhaustive state-space search to achieve exact solutions, with the computational effort growing exponentially or faster with system size. Various kinds of heuristic methods are therefore often used to provide good approximate solutions.

Artificial neural networks (ANN) provide a very versatile method for obtaining approximate solutions to combinatorial optimization problems. In contrast to most other methods, it does not rely on a direct exploration of the discrete state-space. Instead, it is based on using a set of continuous variables, giving a probabilistic representation of the elementary decisions involved in the problem. These variables evolve through a continuous, interpolating space, feeling their way towards a good solution driven by annealing in an artificial temperature.

Early versions were entirely based on binary (Ising) variables for representing elementary decisions, but modern variants typically involve more general multi-state (Potts) variables.

The ANN approach yields high-quality solutions to a variety of problem types, spanning from simple scheduling and assignment problems to complex communications routing. It also has the advantage that the dynamics can be directly implemented in VLSI, allowing for an unusually tight bond between algorithm and hardware. The pure ANN approach can also be extended in various ways to yield methods for approaching more general problem types.

Key elements in the neural approach are the *mean-field* approximation (Hopfield and Tank 1985; Peterson and Söderberg 1989), *annealing*, and for many problems the *Potts* formulation (Peterson and Söderberg 1989). Recently, also *propagator* methods have proven most valuable for handling topological complications (Lagerholm et al. 1997; Häkkinen et al. 1998).

2 The Hopfield Model

Simple models for magnetic systems ("spin glasses") strongly resemble recurrent networks – with the direction of a microscopic elementary magnet seen as analogous to the firing-state of a neuron – and have been a strong source of inspiration for neural network studies.

Recurrent networks first attracted interest in the context of associative memory. The Hopfield model (Hopfield 1982) of an associative memory is based on the energy function

$$E(s_1 \dots s_N) = -\frac{1}{2} \sum_{i,j} w_{ij} s_i s_j \quad , \tag{1}$$

in terms of N binary variables (*Ising* neurons) $s_i = \pm 1$, and a set of symmetric weights w_{ij}, appropriately chosen depending on the patterns to be stored. The patterns will then appear as local minima of $E(s)$. With an asynchronous local optimization dynamics

$$s_i(t+1) = \mathrm{sgn}\left(\sum_{j \neq i} w_{ij} s_j(t) \right) \quad , \tag{2}$$

the state of the system will be attracted to a nearby local minimum representing one of the patterns. Thus the system has the functionality of an associative memory.

3 Optimization with Hopfield Networks

In a similar way, a Hopfield-type energy function (1) can be used to represent an optimization problem (Hopfield and Tank 1985), with a dedicated choice of weights such that an optimal solution to the problem is represented by minimizing $E(s)$. This yields an archetype of the neural approach to optimization.

The performance is enhanced by using a softer dynamics, with the step function $\mathrm{sgn}(\cdot)$ in (2) replaced by the sigmoid $\tanh(\cdot/T)$, with T an artificial temperature. With annealing in T the resulting mean-field (MF) neurons will, as $T \searrow 0$, relax to a stable configuration representing a tentative solution to the problem. The key problem here is to reach the global minimum or at least a very low-lying local minimum.

Below we will discuss this approach in some detail, using the graph bisection (GBis) problem as an example. Systems with generic multi-state (Potts) spins instead of binary Ising spins can be treated in a similar way (Peterson and Söderberg 1989), and will be discussed later.

3.1 The Graph Bisection Problem

The ANN approach is particularly transparent for GBis because of its binary nature (Peterson and Anderson 1988). The problem is defined as follows. A graph of N nodes is to be partitioned into two halves, such that the cut-size (the number of connections between the two halves) is minimal (Fig. 1*A*). The graph defines a symmetric connection matrix J such that $J_{ij} = 1$ if nodes

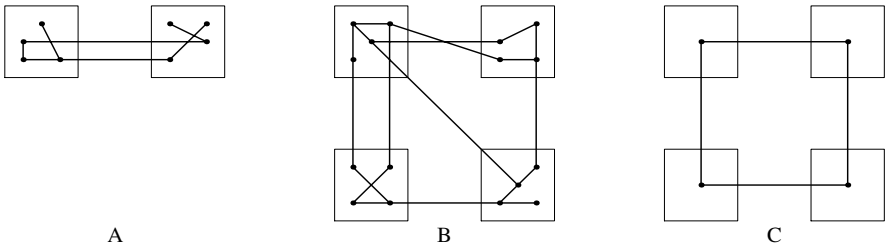

Fig. 1. A, a graph bisection problem. **B**, a $K = 4$ graph partition problem. **C**, an $N = 4$ TSP problem.

i and j are connected, and 0 if they are not (or if they are identical). The distribution of nodes between the two halves can be represented by binary spins as follows. To each node i a binary spin $s_i = \pm 1$ is assigned, representing whether the node winds up in the left or right partition, in terms of Fig. 1*A*.

With this notation, $J_{ij}s_is_j$ will be non-vanishing only if nodes i, j are connected, yielding 1 if they are in the same partition and -1 if not. Thus, apart from a constant, the cut-size is measured by $-1/2\sum_{ij} J_{ij}s_is_j$.

This is not enough for an energy function: The cut-size is trivially minimized by putting all the nodes in the same "half". We need to add a *global constraint term* that penalizes situations where the nodes are not evenly partitioned. Since $\sum s_i = 0$ only for a balanced partition, a term proportional to $(\sum s_i)^2$ will subsequently do the trick. Discarding the constant diagonal part $\sum_i s_i^2 = N$, we can thus represent the problem with a Hopfield energy function (1) with $w_{ij} = J_{ij} - \alpha\,(1 - \delta_{ij})$:

$$E = -\frac{1}{2}\sum_{ij} J_{ij}s_is_j + \frac{\alpha}{2}\left(\left(\sum_i s_i\right)^2 - \sum_i s_i^2\right) \quad , \tag{3}$$

where the constraint coefficient α sets the relative strength of the penalty term.

The generic form of (3) is

$$E = \text{Cost} + \text{Global constraint} \quad , \tag{4}$$

which is characteristic of combinatorial optimization problems. The source of the difficulty inherent in this kind of problem is very transparent here: The system is frustrated by the competition between the two terms, cost and global constraint. This frequently leads to the appearance of many local energy-minima.

The next step is to find an efficient procedure for minimizing E, such that suboptimal local minima are avoided as much as possible.

3.2 Simulated Annealing

Attempting to minimize E according to a local optimization rule (like (2)) will very likely give the result that the system ends up in some suboptimal local minimum close to the starting point.

A better strategy is then to employ a stochastic algorithm that allows for uphill moves. One such method is Simulated Annealing (SA) (Kirkpatrick et al. 1983), in which sequences of configurations are generated with neighborhood search methods, in a way designed to emulate the Boltzmann distribution

$$P[s] = \frac{1}{Z} e^{-E[s]/T} \quad . \tag{5}$$

Here, Z is the *partition function*

$$Z = \sum_{[s]} e^{-E[s]/T} \quad , \tag{6}$$

while $T > 0$ is a artificial temperature representing the noise level of the system. For $T \searrow 0$ the Boltzmann distribution becomes concentrated to the configuration minimizing E. If configurations are generated with a slowly decreasing T – annealing –, they are less likely to get stuck in local minima than if T is set to 0 from the start. The disadvantage of this method is that in order to be reasonably certain to hit the global minimum, one has to employ a very slow annealing schedule, which is very CPU consuming.

3.3 The Mean-Field Approximation

The MF approach aims at approximating the stochastic SA method with a deterministic dynamics, based on the MF approximation, which can be derived as follows. Introduce for each spin s_i a new variable v_i, living in a linear space (in this case $\mathbb{R}$) containing the compact state-space ($\{\pm 1\}$) of the spin, and set it equal to the spin by means of a Dirac delta function. This yields

$$Z = \sum_{[s]} \int d[v] e^{-E[v]/T} \prod_i \delta(s_i - v_i) \quad . \tag{7}$$

Next, Fourier-expand the delta functions in terms of conjugate variables u_i, giving

$$Z \propto \sum_{[s]} \int d[v] \int d[u] e^{-E[v]/T} \prod_i e^{u_i(s_i - v_i)} \quad . \tag{8}$$

Then carry out the (by now trivial) sum over the spins $\{s\}$, and write the resulting product of cosh factors as a sum of logarithms in the exponent:

$$Z \propto \int d[v] \int d[u] e^{-E[v]/T - \sum_i u_i v_i + \sum_i \log\cosh u_i} \quad . \tag{9}$$

The partition function is now rewritten entirely in terms of the new, continuous variables $\{u, v\}$, with an effective energy $E_{\text{eff}}[v,u] = E[v] + T\sum_i (u_i v_i - \log\cosh u_i)$ in the exponent. So far no approximation has been made.

We next make a saddle-point approximation, assuming that the integral in (9) is dominated by an extremal value of E_{eff}. This occurs for $\partial E_{\text{eff}}/\partial v_i = \partial E_{\text{eff}}/\partial u_i = 0$, yielding

$$v_i = \tanh u_i \quad , \tag{10}$$

$$u_i = -\frac{1}{T}\frac{\partial E[v]}{\partial v_i} \quad . \tag{11}$$

Upon elimination of $\{u\}$, we obtain the *MF equations*

$$v_i = \tanh\left(-\frac{1}{T}\frac{\partial E[v]}{\partial v_i}\right) \quad . \tag{12}$$

The solutions $\{v_i\}$, the *MF neurons*, represent approximations to the thermal averages $\langle s_i \rangle_T$ of the original binary spins.

For the Hopfield energy function (1) we obtain

$$v_i = \tanh\left(\frac{1}{T}\sum_j w_{ij} v_j\right) \quad . \tag{13}$$

What we have obtained is obviously a softer version of (2).

3.4 The Mean-Field Dynamics

The MF equations (12) or (13) are solved iteratively, either synchronously or asynchronously, under annealing in T. This yields a deterministic dynamics, characteristic of a recurrent ANN. The resulting neurons $\{v_i\}$ will not be confined to ± 1, but will populate the intervening interval $[-1, 1]$.

The MF dynamics typically exhibits two phases: At high temperatures the sigmoid $\tanh(\cdot/T)$ is very smooth, and the system typically relaxes into a trivial fixed point $v_i^0 = 0$. As the temperature is lowered the sigmoid becomes steeper, and at some critical temperature T_{c} a bifurcation occurs, where v_i^0 becomes unstable and non-trivial fixed points v_i^* emerge. As $T \to 0$, the dynamics of (2) is recovered, and v_i^* are forced to ± 1, representing a specific decision made as to the solution of the problem in question (Fig. 2).

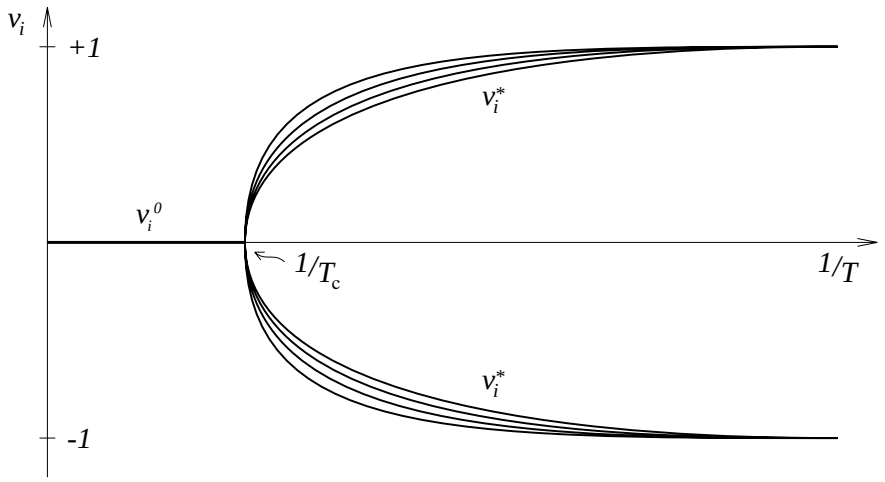

Fig. 2. Fixed points of the binary MF dynamics and the bifurcation, schematically depicted in terms of $\{v_i\}$ as a function of $1/T$.

The position of T_c can be estimated by linearizing (13) around the trivial fixed point $v_i^0 = 0$, i.e. by replacing the sigmoid function (tanh) by its argument. This leads to

$$v_i = \frac{1}{T} \sum_j w_{ij} v_j \tag{14}$$

valid for small $\{v\}$.

For *synchronous* updating it is clear that the trivial fixed point becomes unstable as soon as an eigenvalue λ of the matrix $\mathbf{w}/T$ is > 1 in absolute value. This happens if T equals the largest positive eigenvalue of $\mathbf{w}$ ($\Rightarrow \lambda = 1$), or minus the largest negative one ($\Rightarrow \lambda = -1$), whichever is larger. Prior estimation of the largest eigenvalues is straightforward; apart from determination of T_c, this is also important for avoiding oscillatory behavior, resulting for $\lambda = -1$.

In the preferred case of *serial* updating, the philosophy is the same but the analysis slightly more complicated; the main change (in the absence of diagonal terms) is that the fixed point can only loose stability with $\lambda = 1$, eliminating oscillatory behavior (Peterson and Söderberg 1989). This is why the diagonal terms where removed in (3); as a consequence all self-couplings are eliminated, in the sense that the updated value of v_i does not depend on its old value.

4 Optimization with Potts Neural Networks

For GBis and many other optimization problems, an encoding in terms of binary elementary variables is natural. However, for a generic problem, the

natural elementary decisions are not always binary, but often of the type one-of-K with $K > 2$.

In early attempts to approach such problems by neural network methods, the problem was forced into a binary form by the use of *neuron multiplexing* (Hopfield and Tank 1985): For each elementary K-fold decision, a set of K binary 0/1-neurons was used, with the additional constraint that precisely one of them be on (=1). These *syntactic* constraints were implemented in a soft manner as penalty terms. As it turned out in the original work on the traveling salesman problem (Hopfield and Tank 1985), as well as in subsequent investigations for the graph partition problem (Peterson and Söderberg 1989), this approach does not generally yield high-quality solutions in a parameter-robust way.

An alternative and better encoding results from using *Potts neurons* with the syntactic constraint built in. In this way the dynamics is confined to the relevant parts of the solution space (Fig. 3), leading to a drastically improved performance.

4.1 Potts Spins

A K-state Potts spin should have K possible values (states). For our purposes, the best representation is in terms of a vector of 0/1-components. Thus, denoting a spin by $\mathbf{s} = (s_1, s_2, \ldots, s_K)$, the ath possible state is given by setting the ath component of $\mathbf{s}$ to 1, and the rest to zero. The state vectors point to the corners of a regular K-simplex (see Fig. 3 for the case of $K = 3$), and fulfill by definition the syntactic constraint $\sum_a s_a = 1$.

Potts Mean Field Equations. The MF equations for a system of Potts spins $\mathbf{s}_i$ with a given energy function $E(\mathbf{s})$ are derived following the same path as in the Ising neuron case: Rewrite the partition function as an integral over $\mathbf{u}_i$ and $\mathbf{v}_i$ (which now live in $\mathbb{R}^K$), and make a saddle-point approximation. One obtains

$$u_{ia} = -\frac{\partial E(\mathbf{v})}{\partial v_{ia}}/T \tag{15}$$

$$v_{ia} = \frac{e^{u_{ia}}}{\sum_b e^{u_{ib}}} \tag{16}$$

to be solved by iteration, sequentially in i, with annealing in T.

From (16) it follows that the *Potts neurons* $\mathbf{v}_i$, approximating the thermal averages of $\mathbf{s}_i$, satisfy

$$v_{ia} > 0, \qquad \sum_a v_{ia} = 1 \ . \tag{17}$$

One can think of the neuron component v_{ia} as the *probability* for the ith Potts spin to be in state a. For $K = 2$ one recovers the formalism of the Ising case in a slightly disguised form.

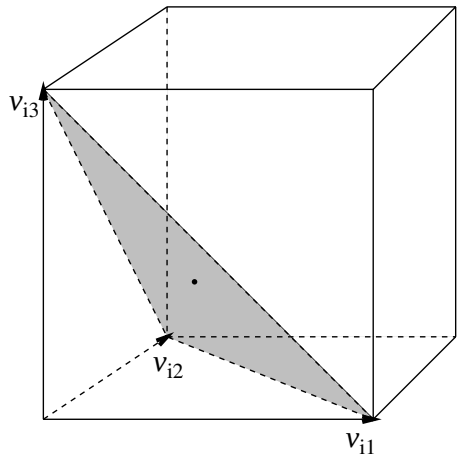

Fig. 3. The volume of solutions corresponding to the neuron multiplexing encoding for $K = 3$. The shaded plane represents the solution space of the corresponding Potts encoding; its corners correspond to the allowed Potts spin states, while the black dot at the center marks the trivial fixed point of no decision.

As in the binary case, there is at high T a trivial fixed point $v_{ia}^0 = 1/K$ corresponding to no decision, and a linear stability analysis yields a critical temperature T_c. For $T \to 0$, the Potts neurons are forced to a decision, with one component approaching unity, and the others zero (see Fig. 3).

4.2 Straightforward Applications

For a large class of combinatorial optimization problems, a straight-forward Potts MF ANN approach can be used, with the following basic steps:

- Map the problem onto a Potts neural network by a suitable encoding of the solution space and an appropriate choice of energy function.
- Compute a suitable starting temperature, e.g. by using a linear stability analysis of the MF dynamics to compute T_c.
- While annealing, solve the MF equations iteratively.
- When the system has settled, the solutions are checked with respect to constraint satisfaction. If needed, one may perform a simple corrective post-processing, or rerun the system (possibly with modified constraint coefficients).

Note that with a Potts encoding there is no difficulty in treating problems having a varying K, in the sense that distinct elementary decisions have different order.

Below, a selected sample of straightforward applications will be briefly discussed, with references to relevant original articles.

The Graph Partition Problem. A generalization of graph bisection is *graph partitioning* (GP): An N-node graph, defined by a symmetric connection matrix $J_{ij} = 0, 1$, $i \neq j = 1, \ldots, N$, is to be partitioned into K subsets of N/K nodes each, while minimizing the cut-size, i.e. the number of connected node pairs winding up in different subsets (see Fig. 1*B*).

This problem is naturally encoded in terms of Potts spins as follows: For each node $i = 1, \ldots, N$, a K-state Potts spin $\mathbf{s}_i = (s_{i1}, \ldots, s_{iK})$ is assigned, where the component s_{ia} being 1 represents the decision that node i should belong to subset a. A suitable energy function is then given by (cf. (3))

$$E = -\frac{1}{2} \sum_{i,j=1}^{N} J_{ij} \mathbf{s}_i \cdot \mathbf{s}_j + \frac{\alpha}{2} \left(\left(\sum_{i=1}^{N} \mathbf{s}_i \right)^2 - \sum_{i=1}^{N} \mathbf{s}_i^2 \right) \quad , \tag{18}$$

where the first term is a cost term (cut-size), while the second is a penalty term with a minimum when the nodes are equally partitioned into the K subsets. Note that the (trivial) diagonal contributions are eliminated in the second term; this stabilizes the dynamics for sequential updating, in analogy to the case for binary neurons (Peterson and Söderberg 1989).

The Traveling Salesman Problem. In the traveling salesman problem (TSP) the coordinates $\mathbf{x}_i \in \mathcal{R}^2$ of a set of N cities are given. A closed tour of minimal total length is to be chosen such that each city is visited exactly once. This problem is somewhat reminiscent of (the trivial) $K = N$ graph partition (see Fig. 1*C*). We define an N-state Potts neuron $\mathbf{s}_i$ for each city $i = 1, \ldots, N$, such that the component s_{ia} $(a = 1, \ldots, N)$ is 1 if city i has the tour number a, and 0 otherwise. Let d_{ij} be the distance between cities i and j. Then a suitable energy function is given by

$$E = \sum_{i,j=1}^{N} d_{ij} \sum_{a=1}^{N} s_{ia} s_{j(a+1)} + \frac{\alpha}{2} \left(\left(\sum_{i=1}^{N} \mathbf{s}_i \right)^2 - \sum_{i=1}^{N} \mathbf{s}_i^2 \right) \quad , \tag{19}$$

where the first term is a cost term, and the second a soft constraint term penalizing configurations where two cities are assigned the same tour number (Peterson and Söderberg 1989; Peterson 1990).

Scheduling Problems. Scheduling problems have a natural formulation in terms of Potts neurons. In its purest form, a scheduling problem has the simple structure: For a given set of events, a time-slot and a location are to be chosen, each from a set of allowed possibilities, such that no clashes occur. Such a problem consists entirely in fulfilling a set of basic no-clash constraints, each of which can be encoded as a *penalty term* that will vanish when the constraint is obeyed (Gislén et al. 1989).

In many realistic scheduling applications, however, there exist additional preferences within the set of legal schedules, that lead to the appearance also

of *cost terms*. In Gislén et al. 1992b, a set of real-world scheduling problems was successfully dealt with.

5 Refinements of the Potts MF Approach

For many optimization problems, complications arise that need special treatment, and the straight-forward method of the previous section will have to be modified or supplemented. In this section we will address a set of such common complications.

5.1 Non-Quadratic Energy Functions

Not all optimization problems can be formulated in terms of a quadratic energy function, even though the state-space can be encoded in terms of a set of Potts neurons. This presents no principal difficulty, and one can still use (15). However, a possible practical problem arises from the induced self-couplings in the energy function (terms with non-linearities in a single spin), that might affect performance (Peterson and Söderberg 1989).

With a quadratic E, self-couplings can be avoided by removing all diagonal terms, $s_{ia}s_{ib} \to \delta_{ab}s_{ia}$. Such a procedure can be generalized to any polynomial E. Although in principle any energy-function of a finite number N of Potts spins can be rewritten as a polynomial of at most degree N, this can be difficult in practice for large N.

An efficient and general method for avoiding self-couplings altogether is to replace the derivative in (15) by a difference:

$$u_{ia} = -\frac{1}{T}\left[E(\mathbf{v})|_{\mathbf{v}_i=\mathbf{e}_a} - E(\mathbf{v})|_{\mathbf{v}_i=\mathbf{0}}\right] \ , \tag{20}$$

where $\mathbf{v}_i = (v_{i1}, \ldots, v_{iK})$, and $\mathbf{e}_a$ is the principal unit-vector in the a-direction. Whenever E is free of self-couplings, (15) and (20) are equivalent.

5.2 Inequality Constraints

In the problems mentioned in the previous section, the constraints considered were all of the *equality* type, $f(s) = 0$, that could be implemented with quadratic penalty terms $\propto f(s)^2$. However, in many optimization problems, in particular those of resource allocation type, one has to deal with *inequalities*. An inequality constraint, $g(s) \leq 0$, can be implemented with a penalty term e.g. proportional to

$$\Phi(g) = g\Theta(g) \ , \tag{21}$$

with Θ the Heaviside step function: $\Theta(x) = 1$ if $x > 0$ and 0 otherwise. Of course, such a non-polynomial term in the energy function must be handled using (20).

A problem category with inequality constraints is the *knapsack problem*, where one has a set of N *items* i with associated *utilities* c_i and *loads* a_{ki}. The goal is to fill a "knapsack" with a subset of the items such that their total utility,

$$U = \sum_{i=1}^{N} c_i s_i \ , \tag{22}$$

is maximized, subject to a set of M load constraints,

$$\sum_{i=1}^{N} a_{ki} s_i \leq b_k, \qquad k = 1, \ldots, M \ , \tag{23}$$

defined by load *capacities* $b_k > 0$. In (Ohlsson et al. 1993), a set of difficult random knapsack problems were approached with a neural method, based on the energy function

$$E = -\sum_{i=1}^{N} c_i s_i + \alpha \sum_{k=1}^{M} \Phi \left(\sum_{i=1}^{N} a_{ki} s_i - b_k \right) \tag{24}$$

in terms of binary spins $s_i \in \{1, 0\}$, representing whether or not item i goes into the knapsack. In (Ohlsson and Pi 1997) this approach was extended to multiple knapsacks and generalized assignment problems, using Potts MF neurons.

5.3 Routing Problems

Many *network-routing* problems can be conveniently handled in a Potts MF approach. We will briefly illustrate the method on the simple shortest-path problem: Given a set of N cities connected by a network of roads, find the shortest path from city a to city b.

Regarding the cities as nodes in a network, and denoting by d_{ij} the length of a direct path, an *arc*, between cities i and j, the problem is equivalent to finding the shortest sequence of arcs leading from node a to b.

This problem can be solved in polynomial time using e.g. the Bellman-Ford (BF) algorithm (Bellman 1958), where every node i estimates its distance D_{ib} to b, minimized with respect to the choice of a continuation node j among its neighbors (nodes directly connected to i via an arc):

$$D_{ib} = \min_j \left(d_{ij} + D_{jb} \right), \quad i \neq b \ , \tag{25}$$

while $D_{bb} = 0$. Iteration of (25) gives convergence in less than N steps, and D_{ab} as well as the path can be directly read off.

By introducing a Potts MF neuron to handle the neighbor choice for each node, we obtain a softer version of the BF algorithm,

$$D_{ib} = \sum_j v_{ij} \left(d_{ij} + D_{jb} \right), \qquad \text{with} \qquad v_{ij} = e^{-(d_{ij}+D_{jb})/T} \ . \tag{26}$$

In the limit of $T \to 0$, a strict minimum is obtained, and BF is recovered. Note that this is an example of a problem where the natural Potts encoding lead to a varying K, since different nodes typically have a different number of neighbors.

In a similar way also more complex routing problems can be formulated in terms of an optimal local neighbor choice, that naturally lends itself to a MF recast. An appealing feature of such an approach is the *locality* inherited from BF: The information needed for the neighbor choice is local to the node and its neighbors.

Note that the basic philosophy here, borrowed from the BF algorithm, is slightly unconventional: Instead of attempting to minimize a global energy function, each node strives to minimize its own, local objective function.

In (Häkkinen et al. 1998) a set of communication routing problems, based on various combinations of unicast and multicast in a network with arcs of finite capacity, were approached along these lines.

The Propagator. Typical of the type of approach sketched above, is the use of *node-node* MF Potts neurons v_{ij} for the local direction of a path from a node to its neighbors.

As a convenient tool for monitoring the global topological properties of fuzzy paths (occurring for $T > 0$), a propagator formalism has been devised (Häkkinen et al. 1998), defined in terms of a propagator matrix $\mathbf{P} = (\mathbf{1} - \mathbf{v})^{-1}$, or

$$P_{ij} = \delta_{ij} + v_{ij} + \sum_k v_{ik} v_{kj} + \sum_{kl} v_{ik} v_{kl} v_{lj} + \ldots \ . \tag{27}$$

In the absence of loops, P_{ij} will be 1 if there exists a path from i to j, and 0 if not (sharp path). For the case of fuzzy paths, the propagator has an obvious probabilistic interpretation. In addition, the propagator will diverge in the presence of loops, and can therefore be used to avoid loop formation.

By using suitable methods for doing the matrix-inversion, required for the computation of the propagator elements, in a clever way, the increase in computational demand can be kept manageable.

As an example of the use of an analogous formalism in a somewhat different setting, see (Lagerholm et al. 1997), where a set of semi-realistic airline-crew scheduling problems was treated, with v_{ij} in (27) controlling the transfer of a crew from a flight i to a connecting flight j.

5.4 Hybrid Approaches

A large class of optimization problems can be viewed as *parametric assignment* problems, containing elements of both discrete assignment and parametric fitting to given data, e.g. using templates with a known structure. The

assignment part can then be encoded in terms of Potts neurons, while the template part may be formulated in terms of a set of continuous, adjustable parameters. This leads to a kind of hybrid approach, *deformable templates*.

Track finding in high energy physics, where circular or spiral-shaped tracks are to be extracted from a set of data points, is an example where this approach is suitable (Ohlsson et al. 1992; Gyulassy and Harlander 1991). In addition, certain pure assignment problems with a well-defined geometric structure can be recast in this form; an example is given by TSP (Durbin and Willshaw 1987).

6 Summary and Discussion

From the above-discussed applications of the Potts MF approach to optimization, the following general picture emerges: For a variety of problem types and sizes, the MF method consistently performs (without excessive fine-tuning) roughly in parity with available comparison methods, typically designed to perform well for the particular problem class. With a prior estimate of T_{C}, convergence is consistently achieved after a very modest number (typically 50–100) of iterations, independently of problem size.

Inequality constraints and other complications leading to non-polynomial energy functions, can be handled in a straight-forward manner.

For communication routing problems and other problems of a topological nature, a Potts approach can be devised, where a propagator formalism enables the monitoring of global topological features, while the dynamics is based on a strictly local information.

For parametric assignment problems, like track-finding, a deformable templates method can be used, utilizing Potts neurons in combination with analog variables; a similar hybrid approach can be applied also to a class of low-dimensional geometrical assignment problems, such as TSP.

Generalizations. A binary (Ising) spin can be considered as a vector living on a "sphere" in one dimension. The MF approach can be generalized to variables defined on spheres in higher dimensions (or indeed in any compact manifold). Such *rotor* neurons can be used in geometrical optimization problems involving angular variables (Gislén et al. 1992a).

The MF approximation can be viewed as a variational approach – the true energy $E(s_i)$ is approximated by a trial one $E_0(s_i; u_i)$ (in this case linear), which is optimized with respect to the variational parameters u_i (the coefficients). This general procedure can be used in a wide range of situations not necessarily confined to discrete optimization problems (see e.g. Jönsson et al. 1993, and references therein, for an application to polymers).

References

Bellman, R. (1958): On a routing problem. Quarterly of Appl. Math. **16**, 87–90

Durbin, R., and Willshaw, D. (1987): An analog approach to the traveling salesman problem using an elastic net method. Nature **326**, 689–691

Gislén, L., Peterson, C., and Söderberg, B. (1989): Teachers and Classes with Neural Networks. Int. J. Neural Syst. **1**, 167–176

Gislén, L., Peterson, C., and Söderberg, B. (1992 a): Rotor neurons – Basic formalism and dynamics. Neural Computat. **4**, 737–745

Gislén, L., Peterson, C., and Söderberg, B. (1992 b): Complex scheduling with Potts neural networks. Neural Computat. **4**, 805–831

Gyulassy, M., and Harlander, H. (1991): Elastic tracking and neural network algorithms for complex pattern recognition. Comput. Phys. Commun. **66**, 31–46

Häkkinen, J., Lagerholm, M., Peterson, C., and Söderberg, B. (1998): A Potts neuron approach to communication routing. Neur. Computat. **10**, 1587-1599

Hopfield, J.J. (1982): Neural networks and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. USA **79**, 2554–2558

Hopfield, J.J., and Tank, D.W. (1985): Neural computation of decisions in optimization problems. Biol. Cybern. **52**, 141–152

Jönsson, B., Peterson, C., and Söderberg, B. (1993): A variational approach to correlations in polymers. Phys. Rev. Lett. **71**, 376–379

Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983): Optimization by simulated annealing. Science **220**, 671–680

Lagerholm, M., Peterson, C., and Söderberg, B. (1997): Airline crew scheduling with Potts neurons. Neur. Computat. **9**, 1589–1599

Ohlsson, M., Peterson, C., and Söderberg, B. (1993): Neural networks for optimization problems with inequality constraints – The knapsack problem. Neural Computat. **5**, 331–339

Ohlsson, M., Peterson, C., and Yuille, A. (1992): Track finding with deformable templates – The elastic arms approach. Comput. Phys. Commun. **71**, 77–98

Ohlsson, M., and Pi, H. (1997): A study of the Mean Field Approach to Knapsack Problems. Neur. Netw. **10**, 263–271

C. Peterson (1990): Parallel distributed approaches to combinatorial optimization problems – Benchmark studies on TSP. Neural Computat. **2**, 261–269

Peterson, C., and Anderson, J. R. (1988): Neural Networks and NP-complete Optimization Problems – A Performance Study on the Graph Partition Problem. Compl. Syst. **2**, 59–89

Peterson, C., and Söderberg, B. (1989): A new method for mapping optimization problems onto neural networks. Int. J. Neural Syst. **1**, 3–22