Improving Convergence and Solution Quality of Hopfield-Type Neural Networks with Augmented Lagrange Multipliers

Stan Z. Li
School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore 639798
E-mail: szli@szli.eee.ntu.edu.sg

Abstract

Hopfield-type networks convert a combinatorial optimization to a constrained real optimization and solve the latter using the penalty method. There is a dilemma with such networks: When tuned to produce good quality solutions, they can fail to converge to valid solutions; when tuned to converge, they tend to give low quality solutions. This paper proposes a new method, called the Augmented Lagrange-Hopfield (ALH) method, to improve Hopfield-type neural networks in both the convergence and the solution quality in solving combinatorial optimization. It uses the augmented Lagrange method, which combines both the Lagrange and the penalty methods, to effectively solve the dilemma. Experimental results on the TSP show superiority of the ALH method over the existing Hopfield-type neural networks in the convergence and solution quality. For the 10-city TSP's, ALH finds the known optimal tour with 100% success rate, as the result of 1000 runs with different random initializations. For larger size problems, it also finds remarkably better solutions than the compared methods while always converging to valid tours.

I. Introduction

Developing neural algorithms for solving combinatorial optimization problems has been one of the main interests in neural network research. An important work in this area is that of Hopfield [1] and Hopfield and Tank [2]. Following that, many related methods are proposed such as the elastic net of Durbin and Willshaw [3] and the mean field annealing of Peterson and Soderberg [4]. Those methods results in algorithms which correspond to certain highly-interconnected networks of non-linear neurons. The appeal is their suitability for analog "silicon implementation" in the form of application specific integrated circuits.

Recent studies have found that the original Hopfield network has convergence and stability problems [5]. In solving the traveling salesman problem (TSP), for example, the Hopfield network either fails to converge to valid tours or gives solutions far from the known optimal tour [5]. The convergence problem also exists in the elastic net approach when the temperature approaches zero [6].

Various modifications are proposed to improve the convergence of the Hopfield network. For example, Brandt *et al.* [7] modified the energy function of Hopfield and Tank [2] to improve the convergence to valid solutions. Protzel *et al.* [8] studied Brandt *et al.* 's formulation with different parameters. A problem is that while the modified versions of the Hopfield-type networks may converge well to valid solutions, they may not converge to good quality solutions [8].

From optimization viewpoint, the Hopfield neural network and the modified versions essentially belong to the penalty method for solving the constrained real optimization into which a combinatorial optimization is converted. In order for the penalty method to converge to a feasible solution, the weighting factors for the penalty terms must be sufficiently large. However, as the penalty terms become stronger, the role of the original objective function becomes relatively weaker. The solutions thus found are affected more by the penalty terms and hence less favorable in terms of the original objective. Worse still, as they becomes larger and larger, the problem becomes ill-conditioned [9]. This is a typical problem with the penalty method and explains why it is difficult to obtain good quality solutions and good convergence simultaneously with the Hopfield-type networks.

This paper proposes a new method, called the Augmented Lagrange-Hopfield (ALH) method, which uses the augmented Lagrange multiplier method [10], [11], [12] to overcome the problems with the the Hopfield-type neural networks in solving combinatorial optimization. As in the Hopfield-type methods, a combinatorial optimization is converted to a constrained real optimization. The augmented Lagrange method, which combines both the Lagrange and the penalty methods, effectively overcomes the problems associated with the penalty method or the Lagrange method when used alone. With the Lagrangian multipliers, the constraints are satisfied *exactly* without the need to send the penalty terms to infinity. This not only avoids the ill-conditioning problem but also reduces the unfavorable influence of the penalty terms on the solution quality. With the penalty terms, the zigzagging problem in the standard Lagrange multiplier method is alleviated.

The efficacy of the ALH is demonstrated with the TSP, one of the hardest combinatorial problems, in comparison with the improved Hopfield methods of Brandt *et al.* [7] and and Protzel *et al.* [8]. The ALH method not only produces valid tours all the time in our tests but also yields better quality solutions than the compared methods. For example, for the 10-city TSP's we tested, including one examined in [2], [5], it finds the known optimal tour with 100% success rate, as the result of 1000 runs with different random initializations. For larger size problems, it also finds remarkably better solutions. Although in this paper it is demonstrated using the TSP, the ALH method can well be applied to other combinatorial optimization problems. Successful applications have also been reported for graph matching [13] and image restoration and segmentation [14].

The rest of the paper is organized as follows: Section 2 briefly describes the formulations of the Hopfield type networks. Section 3 describes the ALH method and discusses its relations with other methods. Section 4 evaluates the performance of the ALH in solving the TSP. Conclusions are given in Section 5.

II. Previous Formulations

In neural methods for the TSP, the original combinatorial problem is converted to a form suitable for continuous computation. A convenient way is to consider it as a labeling assignment problem. Let $\mathcal{S} = \{1, \ldots, m\}$ indexes a set of m positions and $\mathcal{L} = \{1, \ldots, M\}$ does a set of $M = m$ cities. We use an M position vector $p_i = [p_i(I) \mid I \in \mathcal{L}]$ to represent the state of the assignment for $i \in \mathcal{S}$. The real value $p_i(I) \in [0, 1]$ reflects the strength with which i is assigned label I. The matrix $p = [p_i(I) \mid i \in \mathcal{S}, I \in \mathcal{L}]$ is the state of the assignment.

Originally, Hopfield and Tank [2] propose the following energy function for the m city TSP

$$\begin{aligned} E_{HT}(p) \;=\; & \frac{A}{2}\sum_i \sum_{j\neq i} \sum_I p_i(I)p_j(I) + \frac{B}{2}\sum_i \sum_I \sum_{J\neq I} p_i(I)p_i(J) \\ & +\frac{C}{2}\left(\sum_i \sum_I p_i(I) - m\right)^2 + \frac{D}{2}\sum_i \sum_I \sum_{J\neq I} d_{IJ}\; p_i(I)\,[p_{i+1}(J) + p_{i-1}(J)] \end{aligned} \tag{1}$$

where d_{IJ} is the distance between cities I and J, and A, B, C, and D are some constants. The variables $p_i(I) \in (0, 1)$ represents the neural output for city I at position i (positions $m+1$ is the same as 1). They relate to internal variables $u_i(I)$ via $p_i(I) = 1/(1 + \mathrm{e}^{-u_i(I)/T})$ where T is a parameter called the temperature. When $T \to 0^+$, all $p_i(I)$ are forced toward corners of the hypercube, where every $p_i(I)$ takes a value of either 1 or 0. The first three terms are constraints for creating valid tours and the last is the measure of the tour length. A minimum $p^* = \arg\min_p E(p)$ is a solution to the problem.

Wilson and Pawley [5] re-examined the formulation and encountered difficulties in converging to valid tours. They found that the solutions represented by local minima may correspond to invalid tours. To remedy this problem, Brandt *et al.* [7] proposed a modified energy

$$\begin{aligned} E_{Brandt}(p) \;=\; & \frac{A}{2}\sum_i \left(\sum_I p_i(I) - 1\right)^2 + \frac{B}{2}\sum_I \left(\sum_i p_i(I) - 1\right)^2 \\ & +\frac{C}{2}\sum_i \sum_I p_i(I)(1 - p_i(I)) + \frac{D}{2}\sum_i \sum_I \sum_{J\neq I} d_{IJ}\; p_i(I)\,[p_{i+1}(J) + p_{i-1}(J)] \end{aligned} \tag{2}$$

Although this gives better convergence to invalid tours than Hopfield and Tank's formulation, the tour quality is not so good as the original one [8].

This is not surprising because Hopfield network is an instance of the penalty method. In order for the penalty method to converge to a feasible solution, the weighting factors for the penalty terms must be sufficiently large. However, as the penalty terms become stronger, the constraints on the original problem becomes relatively weaker and solution quality is deteriorated. Worse still, as they becomes larger and larger, the problem becomes ill-conditioned [9]. This is a dilemma in the penalty method. The ALH method describe in the following is aimed to help the Hopfield-type neural networks converge to good quality solutions.

III. The Augmented Lagrange-Hopfield Method

A. Constrained Minimization Representation

First, the original combinatorial problem is converted into the following constrained minimization

$$\min_p \quad E(p) = \frac{1}{2}\sum_i \sum_I \sum_{J\neq I} d_{IJ}\; p_i(I)\,[p_{i+1}(J) + p_{i-1}(J)] \tag{3}$$

$$\text{subject to} \quad C_k(p) = 0 \qquad k = 1, \dots K \tag{4}$$

$$p_i(I) \geq 0 \qquad \forall i \in \mathcal{S}, \forall I \in \mathcal{L} \tag{5}$$

where $E(p)$ is the objective function of the total tour length; C_k's, where K is an integer number, are some real functions which represent some equality constraints and take the value of zero when the constraints are satisfied. The final solution p^* is subject to additional constraints

$$p_i^*(I) = 0 \quad \text{or} \quad 1 \qquad \forall i, I \tag{6}$$

such that the final solution p^* is in a corner of the hypercube $[0,1]^{\mathcal{S}\times\mathcal{L}}$.

We use the following equality constraints which are some of those used in Hopfield and Tank (1985) and Brandt *et al.* (1988)

$$C_1^i(p) = \sum_I p_i(I) - 1 = 0 \qquad \forall i \tag{7}$$

$$C_2^I(p) = \sum_i p_i(I) - 1 = 0 \qquad \forall I \tag{8}$$

$$C_3^{i,I}(p) = \sum_{j\neq i} p_i(I)p_j(I) = 0 \qquad \forall i, I \tag{9}$$

$$C_4^{i,I}(p) = \sum_{J\neq I} p_i(I)p_i(J) = 0 \qquad \forall i, I \tag{10}$$

$$C_5^{i,I}(p) = p_i(I)(1 - p_i(I)) = 0 \qquad \forall i, I \tag{11}$$

As such, there are a total number of

$$K = 2m + 2m \times m + m \times m = 3m^2 + 2 \times m \tag{12}$$

equality constraints. They can be classified into into three categories: the first consists of the C_1 and C_2 terms, the second C_3 and C_4's, and the third C_5's.

Instead of simply adding the C functions into the tour length function as the penalty terms as in (1) and (2), we use the Augmented Lagrange-Hopfield method, which combines the augmented Lagrange method and the Hopfield network, to solve the above constrained optimization.

B. Solving the Constrained Minimization

As in the Hopfield approach, we introduce the internal variables $u_i(I) \in (-\infty, +\infty)$ $(\forall i, I)$ and relate them to $p_i(I)$ via

$$p_i(I) = \psi_T(u_i(I)) \tag{13}$$

where $\psi_T(x)$ is a sigmoid function controlled by a temperature parameter $T > 0$

$$\psi_T(x) = 1/[1 + e^{-x/T}] \tag{14}$$

With the introduction of the internal u variables, the energy function can be considered as $E(u) = E(p(u))$. This treatment confines $p_i(I)$ to the range $(0, 1)$ to impose the inequality constraints of (5). When $T \to 0^+$, $p_i(I)$ is forced to be 0 or 1 depending on whether $u_i(I)$ is positive or negative, thus imposing the unambiguity constraints of (6).

Next, we use the Lagrange method to impose the equality constraints of (4). Define the Lagrange function of the following form

$$L(p, \gamma) = E(p) + \sum_k \gamma_k C_k(p) \tag{15}$$

where γ_k are called the Lagrange multipliers. In solving the TSP with all the equality constraints in (7) – (11), the complete form $L(p, \gamma)$ is

$$L(p, \gamma) = E(p) + \sum_i \gamma_1^i C_1^i(p) + \sum_I \gamma_2^I C_2^I(p) + \sum_{i,I} \sum_{k=3}^{5} \gamma_k^{i,I} C_k^{i,I}(p) \tag{16}$$

It is a function of the m^2 variables of p and the K variables of γ. For p^* to be a local minimum subject to the constraints, it is necessary that (p^*, γ^*) be a stationary point of the Lagrange function:

$$\begin{array}{c} \nabla_p L(p^*, \gamma^*) = 0 \\ \nabla_\gamma L(p^*, \gamma^*) = 0 \end{array} \tag{17}$$

where ∇_x is the gradient with respect to x. If (p^*, γ^*) is a saddle point for which

$$L(p^*, \gamma) \leq L(p^*, \gamma^*) \leq L(p, \gamma^*) \tag{18}$$

then p^* is a local minimum of $E(p)$ satisfying $C_k(p^*) = 0$, *i.e.* a local solution to the constrained optimization problem [15].

The following dynamic equations can be used to find such a saddle point

$$\frac{\mathrm{d}p_i(I)}{\mathrm{d}t} = -\frac{\partial L(p, \gamma)}{\partial p_i(I)} \tag{19}$$

$$\frac{\mathrm{d}\gamma_k}{\mathrm{d}t} = +\frac{\partial L(p, \gamma)}{\partial \gamma_k} \tag{20}$$

It performs energy descent on p but ascent on γ. This is also called the basic differential multiplier method in [12]. The convergence of this system is illustrated in [16].

The Lagrangian (15) can be augmented by adding penalty terms $[C_k(p)]^2$, giving an augmented Lagrange function [10], [11]

$$L_\beta(p, \gamma) = E(p) + \sum_k \gamma_k C_k(p) + \frac{1}{2} \sum_k \beta_k [C_k(p)]^2 \tag{21}$$

where $\beta_k > 0$ are finite weighting factors. The introduction of the quadratic term $\sum_k \beta_k [C_k(p)]^2$ with $C_k(p) = 0$ does not alter the location of the saddle point. The quadratic terms effectively stabilize the system.

The dynamic equations for finding a saddle point of L_β are

$$\frac{\mathrm{d}p_i(I)}{\mathrm{d}t} = -\frac{\partial L_\beta(p, \gamma)}{\partial p_i(I)} = -\frac{\partial E(p)}{\partial p_i(I)} - \sum_k \gamma_k \frac{\partial C_k(p)}{\partial p_i(I)} - \sum_k \beta_k C_k(p) \frac{\partial C_k(p)}{\partial p_i(I)} \tag{22}$$

$$\frac{\mathrm{d}\gamma_k}{\mathrm{d}t} = +\frac{\partial L_\beta(p, \gamma)}{\partial \gamma_k} = +C_k(p) \tag{23}$$

This corresponds to the modified differential multiplier method [12]. Our experiments show that the penalty terms are necessary for damping the oscillations of the standard Lagrange method and hence helping the convergence of the numerical computation.

In the ALH method, the updating of label assignment is performed on u, rather than on p. Equ.(22) is replaced by

$$\frac{\mathrm{d}u_i(I)}{\mathrm{d}t} = -\frac{\partial L_\beta(p(u),\gamma)}{\partial p_i(I)}\frac{\partial p_i(I)}{\partial u_i(I)}$$

Note that

$$\frac{\partial p_i(I)}{\partial u_i(I)} = \frac{\mathrm{e}^{-u_i(I)/T}}{T(1+\mathrm{e}^{-u_i(I)/T})^2}$$

is always positive. We update u according to

$$\frac{\mathrm{d}u_i(I)}{\mathrm{d}t} = -\frac{\partial L_\beta(p(u),\gamma)}{\partial p_i(I)} \tag{24}$$

This is the method used in the graded Hopfield network [1].

For the TSP,

$$\frac{\partial E}{\partial p_i(I)} = \sum_{J\neq I} d_{IJ}\left[p_{i+1}(J) + p_{i-1}(J)\right]$$

in (22) and there are K Lagrange multipliers γ_k, where K is given in (12), and three different β_k constants for the three categories of the constraints in (7)-(11). The corresponding neural network is composed of the m^2 $p_i(I)$ neurons for the tour state, which are associated with the internal u variables via the sigmoid function circuits, and an additional number of $5m^2+m$ γ neurons for the Lagrangian multipliers.

Numerically, the ALH algorithm implements the dynamics described by (24), (13) and (23), in the following order

$$u_i^{(t+1)}(I) \leftarrow u_i^{(t)}(I) - \mu\left\{q_i^{(t)}(I) + \textstyle\sum_k \gamma_k^{(t)}\frac{\partial C_i(p)}{\partial p_i(I)} + \sum_k \beta_k C_k(p^{(t)})\frac{\partial C_k(p^{(t)})}{\partial p_i^{(t)}(I)}\right\} \tag{25}$$

$$p_i^{(t+1)}(I) \leftarrow \psi_T(u_i^{(t+1)}(I)) \tag{26}$$

and

$$\gamma_k^{(t+1)} \leftarrow \gamma_k^{(t)} + \beta_k C_k(p^{(t+1)}) \tag{27}$$

In the above, μ is a step size factor; and during the update, T may be decreased and β_k increased to speedup convergence. The updating is performed synchronously in parallel for all i and I. Comparing the ALH algorithm with the mean field theoretical algorithm of [4], we see that the ALH does not need the normalization operation required by the latter algorithm and thus is more convenient for analog implementation.

So far, the problem has been formulated for minimizing the energy function $E(p)$. To maximize a gain function $G(p)$ with the constraints as in some applications, we can simply let $E(p) = -G(p)$ and then apply the same updating equations.

C. Remarks

The ALH method reduces to the standard penalty or the Lagrange methods when either γ or β variables are set to zero. . When $\gamma_k = 0$ $(\forall k)$, $L_\beta(p,\gamma) = E(p) + \sum_k \beta_k[C_k(p)]^2$ and the augmented Lagrange method reduces to the penalty method to which the Hopfield network belongs. To approximate the constrained minima of $E(p)$ with $C_k(p) = 0$ by the unconstrained minima of $E(p) + \sum_k \beta_k[C_k(p)]^2$, it is necessary to send β_k to ∞. In numerical computation, however, minimizing $E(p) + \sum_k \beta_k[C_k(p)]^2$ becomes increasingly difficult as $\beta_k \to \infty$. This is due to the ill-conditioning of the problem for large β [9]. Oscillation of u variables is found in the penalty method of [5]. When $\beta_k = 0$, $L_\beta(p,\gamma)$ reduces to the standard Lagrange method of (15). In this simple form, the constraints can prevent convergence to an optimum because of a so-called zigzagging problem [9].

The augmented Lagrange method, which combines both the Lagrange and the penalty methods, effectively overcomes the problems associated with the penalty method or the Lagrange method when used alone. From the viewpoint of duality theory, it is simply the standard Lagrange method for the constrained optimization with $E(p)$ in (3) replaced by $E(p)+\sum_k \beta_k [C_k(p)]^2$. The latter has the same constrained minima as the former when the constraints $C_k(p)=0$ are satisfied. When β_k are sufficiently large, the Hessian of $L_\beta(p,\gamma)$ will be positive definite at a saddle point (p^*,γ^*) [16]. This "convexifies" the problem and thus p^* is a local minimum of $E(p)$ with $C(p^*)=0$ [16], [17]. This method is presented in [12] as the modified differential multiplier method. With the augmented Lagrangian, it is not necessary to send β_k to ∞ in order to arrive at a valid solution. In other words, the β_k values required by the augmented Lagrangian are much smaller than those required by the penalty method. Thus the ill-conditioning caused by large β values is avoided. Furthermore, the convergence rate of the augmented Lagrange method is considerably better than the penalty method given the same set of parameter values.

Platt and Barr [12] apply the Arrow-Hurwicz differential equations to solve the traveling salesman problem formulated in terms of an elastic net [3]. Wacholder *et al.* (1989) combine Platt and Barr's basic differential model, which is without the stabilizing quadratic terms, with the Hopfield method [1]. Although the zigzagging problem was not mentioned there, we find it generally a problem for the quadratic assignment problems such as the TSP.

It is suggested in [18] that a damping term $u_i(I)/\tau$, where $\tau>0$, be added to (24) to force $p_i(I)$ to approach 0 or 1, as in the graded Hopfield neural networks [1]. Our experience is that results are of better quality without this term. In our model, the satisfaction of (4) and (5) with a small enough T in $p_i(I)=1/[1+e^{-u_i(I)/T}]$ is enough for the convergence to (6). The competitive mechanism in (7) and (8) will make the winner take all. In practice, T needs not be very low. We set $T=10^7$ in our experiments and obtained good convergence.

IV. Performance Evaluation

In the following experiments, we evaluate performance of the ALH algorithm through the solving of the TSP. The solution quality is measured in terms of the minimized tour length. The following algorithms are compared:

1. the ALH algorithm (referred to as the ALH),
2. the improved Hopfield algorithm of [7] with the original set of parameters (referred to as the OB), and
3. the improved Hopfield algorithm of [7] with the improved set of parameters suggested in [8] (referred to as the IB).

Each test case is run for 1000 times with different initializations. Additional N runs are re-performed if N invalid tours are generated. So each statistic is based on the results of 1000 valid tours produced.

A. The algorithms

Writing equations (25)–(27) out for the TSP with the constraints (7)–(11) (please refer also to Eq.(16)), we have the following ALH algorithm

$$u_i^{(t+1)}(I) \leftarrow u_i^{(t)}(I)-\mu\left\{D\sum_{J\neq I} d_{IJ}\left[p_{i+1}^{(t)}(J)+p_{i-1}^{(t)}(J)\right]+\frac{\partial E_L}{\partial p_i^{(t)}(I)}+\lambda\frac{1}{2}\frac{\partial E_P}{\partial p_i^{(t)}(I)}\right\} \tag{28}$$

$$p_i^{(t+1)}(I) \leftarrow \psi_T(u_i^{(t+1)}(I)) \tag{29}$$

$$\gamma_k^{(t+1)} \leftarrow \gamma_k^{(t)}+\lambda\beta_k C_k(p^{(t+1)}) \tag{30}$$

In the above, μ is a step size constant, D is a weighting constant, T is a temperature constant, and λ is a non-decreasing factor for the penalty terms. The two partial derivatives in (28) are

$$\frac{\partial E_L}{\partial p_i(I)} = \gamma_1^{i\,(t)}+\gamma_2^{I\,(t)}+\gamma_3^{i,I\,(t)}\sum_{j\neq i} p_j(I)+\gamma_4^{i,I\,(t)}\sum_{J\neq I} p_i(J)+\gamma_5^{i,I\,(t)}(0.5-p_i(I)) \tag{31}$$

which is due to the Lagrange multipliers, and

$$\begin{aligned}\frac{1}{2}\frac{\partial E_P}{\partial p_i(I)} &= \beta_1(\sum_I p_i(I)-1)+\beta_2(\sum_i p_i(I)-1)+ \\ & p_i(I)*\left(\sum_{j\neq i} p_j(I)\right)^2 + p_i(I)*\left(\sum_{J\neq I} p_i(J)\right)^2 + \qquad (32)\\ & p_i(I)*(1-p_i(I))*(0.5-p_i(I)) \qquad (33)\end{aligned}$$

which is due to the penalty terms. Note that the β_k constants are not dependent on i and I.

The dynamic equation for the u variables in Brandt *et al.* 's system is

$$\begin{aligned}u_i^{(t+1)}(I) &\leftarrow u_i^{(t)}(I)-\mu\{A\left(\sum_I p_i^{(t)}(I)-1\right)+B\left(\sum_i p_i^{(t)}(I)-1\right) \qquad (34)\\ & +C(0.5-p_i^{(t)}(I))+D\sum_{J\neq I} d_{IJ}\left[p_{i+1}^{(t)}(J)+p_{i-1}^{(t)}(J)\right]\}\end{aligned}$$

Their p-u relationship is

$$p_i^{(t+1)}(I) \leftarrow \frac{1}{1+\mathrm{e}^{-[u_i^{(t+1)}(I)-0.5]/T}} \qquad (35)$$

noting that there is an offset of 0.5 from $u_i^{(t+1)}(I)$.

In both algorithms, the updating is performed synchronously in parallel for all i and I.

B. Settings for Experiments

The parameters for the ALH equations (28) – (30) are set as follows:

$$\beta_1=\beta_2=100,\ \beta_3=\beta_4=1,\ \beta_5=0, D=10^6, T=10^7, \mu=100 \qquad (36)$$

and λ is increased from 1 to 100 according to $\lambda \leftarrow 1.01\lambda$. The parameter values in Brandt *et al.* 's original implementation are

$$A=B=2,\ C=4,\ D=1,\ T=10,\ \mu=0.1 \qquad (37)$$

Those suggested by Protzel etal [8] are

$$A=B=5,\ C=4,\ D=3,\ T=10,\ \mu=0.005 \qquad (38)$$

Because a tour is circular, we fix the first visited city as city $I=1$ (*i.e.* fix $p_1(1)=1$ and $p_1(1)=0$ for $I\leq 2$). Therefore, the updating is perform for $i,I=2,\ldots,m$. In every run, the p variables are randomly initialized around the center of the unit hypercube as suggested by Brandt *et al.* , that is

$$p_i^{(0)}(I)=0.5+\delta \qquad i,I=2,\ldots,m \qquad (39)$$

where δ is a random number uniformly distributed in $[-10^6,10^6]$. The $u^{(0)}$ variables are determined using a respective u-p relationship. Thus, points scatter around the center. In addition, for the ALH algorithm, all the Lagrange multipliers $\gamma_k^{(0)}$ are initially set to zero.

The iterative process is considered to have been converged if the following three conditions are met:

$$\|p^{(t)}-p^{(t-1)}\|_\infty < 0.01/m \qquad (40)$$

(it means that the state is *stabilized*)

$$|\sum_I p_i(I)-1|<0.01 \quad \text{and} \quad |\sum_i p_i(I)-1|<0.01 \qquad (41)$$

(it means that the state is a *feasible* solution) and

$$1 - p_i(I) < 0.01 \quad \text{or} \quad p_i(I) < 0.01 \tag{42}$$

(it means that the state is *unambiguous*, *i.e.* at a corner of the hypercube.) If any of these (mostly probably the last two) is not satisfied after a certain number of iterations, it is considered as *time-out*.

C. Results and Discussions

First we experiment on three 10-city problems where the city coordinates are shown in Table I. The first data set is the one used in [2], [5] and the other two sets are randomly generated within the unit box. The optimal tours for these data sets are known to be $(1, 4, 5, 6, 7, 8, 9, 10, 2, 3)$, $(1, 4, 2, 9, 7, 5, 10, 6, 3, 8)$ and $(1, 7, 4, 2, 5, 6, 10, 3, 9, 8)$, respectively (the reverse tour being considered as the same tour), with the corresponding minimal tour lengths being 2.691, 3.309 and 3.217.

The tour length histograms for the three algorithms are shown in Fig.1. In terms of the solution quality, the ALH found the minimal tour all the time for the 1000 runs, that is, all the ALH solutions for the 10-city problems were optimal; in comparison, the OB found the minimal tour 8, 31 and 12 times; the IB did 63, 358 and 130 times. In terms of the convergence, all the solutions generated by the ALH were valid tours; in comparison, the OB generated 1 invalid tour for data set 2, and the IB generated 27, 25 and 16 invalid tours for the three data sets, respectively. The invalid solutions were discarded and the same number of runs were re-performed to get the final histogram statistics.

The ALH and the OB converge to stable (*cf.* Equ.(40)), feasible (*cf.* Equ.(41)) and unambiguous (*cf.* Equ.(42)) states all the time; whereas the IB did not converge to a feasible and unambiguous state at all though it did converge to a stable state. All the IB runs were terminated upon time-out. When this happened, the solution had to be post-processed by using the maximum selection operation. However, the corner point thus obtained was not necessarily a valid tour.

Concerning the convergence rate, the OB was the fastest; the IB was the next but with a considerable number of time-outs; and the ALH converged about two to three order slower than the OB. This is shown in detail in Fig.2. Table II summarizes the overall performance.

Does the initialization affects the results? The answer is yes because all the tested methods are local optimizers. We tried a different initialization scheme suggested by Hopfield and Tank [2] (see also [5]), in which each $u_i(I)$ was set to $-T\log(m-1)$ with a small random number $\delta \in [-0.1\ u_i(I), 0.1\ u_i(I)]$ added. As the results, the numbers of the minimal tours found by the ALH were 983, 980 and 999, respectively for the three data sets; those found by IB were 30, 293 and 348; the best tour lengths found by the OB were 2.752, 4.334, and 3.281, none of which were the shortest. This suggests that the initialization scheme is not a good one.

Next we illustrate the results for 16, 22, and 48 city problems. The city coordinates for these three larger size problems were the data sets "ulysses16.tsp", "ulysses22.tsp" and "att48.tsp" given in the TSP library [19], with the optimal tours also given there. We isotropically scaled the coordinates of the cities so that the coordinates (x_i, y_i) were lowered bounded by 0 and upper bounded by 1. Hence, the parameters used for the 10-cite problems needed not be changed for these larger-size problems. After the scaling the three minimal tour lengths were 2.36316, 2.41279 and 4.32452, respectively.

The results are illustrated in Figs.3 and 4. The OB yielded in 0, 1 and 36 invalid tours for the three cases, respectively, and the IB did 11, 90, and 717 times. The ALH not only converged to valid tours all the time but also found tours of much better quality than OB and IB. Table III summarizes the overall performance for these larger size problems.

One thing out of our expectation was that with the 48-city data set, the ALH converged faster and unlike with the 10-, 16- and 22-city data sets, the IB sometimes converged. Our explanation is that the size m is also a parameter which affects the convergence and the solution. When m is larger but the other parameters remain the same, the effect of the parameters is changed. This happened to give the fast convergence.

V. Conclusion

The Augmented Lagrange-Hopfield (ALH) method has been proposed for solving combinatorial optimization problems. Using the augmented Lagrange multipliers, the ALH method overcomes the instability with the penalty method and the standard Lagrangian when used alone, and hence improves convergence. In comparison with the Hopfield-type type neural networks [2], [7], [8], the ALH method not only converges to valid solutions but also produces better quality solutions. It well solves the conflict between the solution quality and the convergence which has been a major problem with the Hopfield-type neural networks.

The ability of the ALH to find better quality solutions than the penalty method of the Hopfield-type networks while maintaining good convergence is due to the use of the augmented Lagrange technique. Basically, one is converting a combinatorial optimization to a constrained real optimization, and solving the constrained problem using an unconstrained real optimization method. In the penalty method, such as the OB and the IB, the objective for the unconstrained optimization is a weighted sum of the original objective function and other terms which penalize the violation of the constraints. To produce valid solutions, the relative weight for the original objective function must be kept relatively small in order to verify the constraints; therefore the solutions are significantly biased from the true minimum of the original objective function, owing to the dominant influence from the penalty terms of constraints.

With the use of Lagrange multipliers as in the augmented Lagrangian, the weighting values for the penalty terms can be much smaller than those required by the penalty method; thus the relative weight for the original objective is increased, which helps yielding a better objective value. At the same time, the augmented Lagrange method also overcomes the zigzagging problem with the standard Lagrange method and improves the convergence. These advantages of the augmented Lagrangian plus the use of Hopfield method for imposing the inequality and unambiguity constraints have made the success of the ALH method.

Although the ALH method always converge, it requires more iterations to do. This drawback may be insignificant with "silicon implementation". The ALH network is more complicated than the Hopfield type networks because of the additional Lagrangian neurons. However, it is these auxiliary neurons that help to achieve the superiority of the ALH method.

The following are some topics for future research: Under what conditions is the Hessian matrix for the ALH energy positive definite? How are they related to the involved parameters? Answering these will lead to the further understanding of the ALH method.

References

[1] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two state neurons", *Proceedings of National Academic Science, USA*, vol. 81, pp. 3088–3092, 1984.

[2] J. J. Hopfield and D. W. Tank, " 'Neural' computation of decisions optimization problems", *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[3] R. Durbin and D. Willshaw, "An analog approach to the travelling salesman problem using an elastic net method", *Nature*, vol. 326, pp. 689–691, 1987.

[4] C. Peterson and B. Soderberg, "A new method for mapping optimization problems onto neural networks", *International Journal of Neural Systems*, vol. 1, pp. 3–22, 1989.

[5] G. V. Wilson and G. S. Pawley, "On the stability of the travelling salesman problem algorithm of Hopfield and Tank", *Biological Cybernetics*, vol. 58, pp. 63–70, 1988.

[6] Martin W. Simmen, "Parameter sensitivity of the elastic net approach to the traveling salesman problem", *Neural Computation*, vol. 3, pp. 363–374, 1991.

[7] R. D. Brandt, Y. Wang, A. J. Laub, and S. K. Mitra, "Alternative networks for solving the travelling salesman problem and the list-matching problem", *in IEEE International Conference on Neural Networks*, vol. 2, pp. 333–340, San Diego 1988, 1988. IEEE, New York.

[8] P. W. Protzel, D. L. Palumbo, and M. K. Arras, "Performance and fault-tolerance of neural networks for optimization", *IEEE Transactions on Neural Networks*, vol. 4, pp. 600–614, 1993.

[9] R. Fletcher, *Practical Methods of Optimization*, Wiley, 1987.

[10] M. J. D. Powell, "A method of nonlinear constraints in minimization problems", in R. Fletcher, editor, *Optimization*, London, 1969. Academic Press.

[11] M. R. Hestenes, "Multipler and gradient methods", *Journal of Optimization Theory and Applications*, vol. 4, pp. 303–320, 1969.

[12] J. C. Platt and A. H. Barr, "Constrained differential optimization", *in Proceedings of the IEEE 1987 NIPS conference*, 1988.

[13] S. Z. Li, "Relaxation labeling using Lagrange multipliers and Hopfield network", *in Proceedings of IEEE International Conference on Image Processing*, vol. 1, pp. 266–269, Washington, D.C., 23-26 October 1995.

[14] S. Z. Li, C. L. Chan and H. Wang, "A constrained optimization method for bayesian image restoration and segmentation", *in Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.1-6, San Francisco, CA, June 16-20 1996.

[15] B. S. Gottfried, *Introduction to Optimization Theory*, Prentice-Hall, 1973.

[16] K. J. Arrow, L. Hurwicz, and H. Uzawa, *Studies in Linear and Nonlinear Programming*, Stanford University Press, 1958.

[17] Dimitri P. Bertsekas, *Constrained optimization and Lagrange multiplier methods*, Academic Press, New York, 1982.

[18] E. Wacholder, J. Han, and R. C. Mann, "A neural network algorithm for the multiple traveling salesman problem", *Biological Cybernetics*, vol. 61, pp. 11–19, 1989.

[19] Bob Bixby and Gerd Reinelt, *Traveling Salesman Problem Library*, ftp://softlib.cs.rice.edu/pub/tsplib/tsblib.tar, 1990.

TABLE I
City coordinates for the Three 10-City Problems

i	X	Y	X	Y	X	Y
1	0.4000	0.4439	0.9721	0.2129	0.6306	0.9947
2	0.2439	0.1463	0.3809	0.0496	0.0117	0.4083
3	0.1707	0.2293	0.7711	0.7435	0.8030	0.4054
4	0.2293	0.7610	0.5861	0.0075	0.4282	0.7175
5	0.5171	0.9414	0.0143	0.8361	0.6192	0.3298
6	0.8732	0.6536	0.6099	0.9089	0.6307	0.1807
7	0.6878	0.5219	0.0263	0.4576	0.2787	0.9442
8	0.8488	0.3609	0.8229	0.5342	0.7513	0.7223
9	0.6683	0.2536	0.2911	0.1300	0.9192	0.6606
10	0.6195	0.2634	0.4203	0.5241	0.7818	0.0525

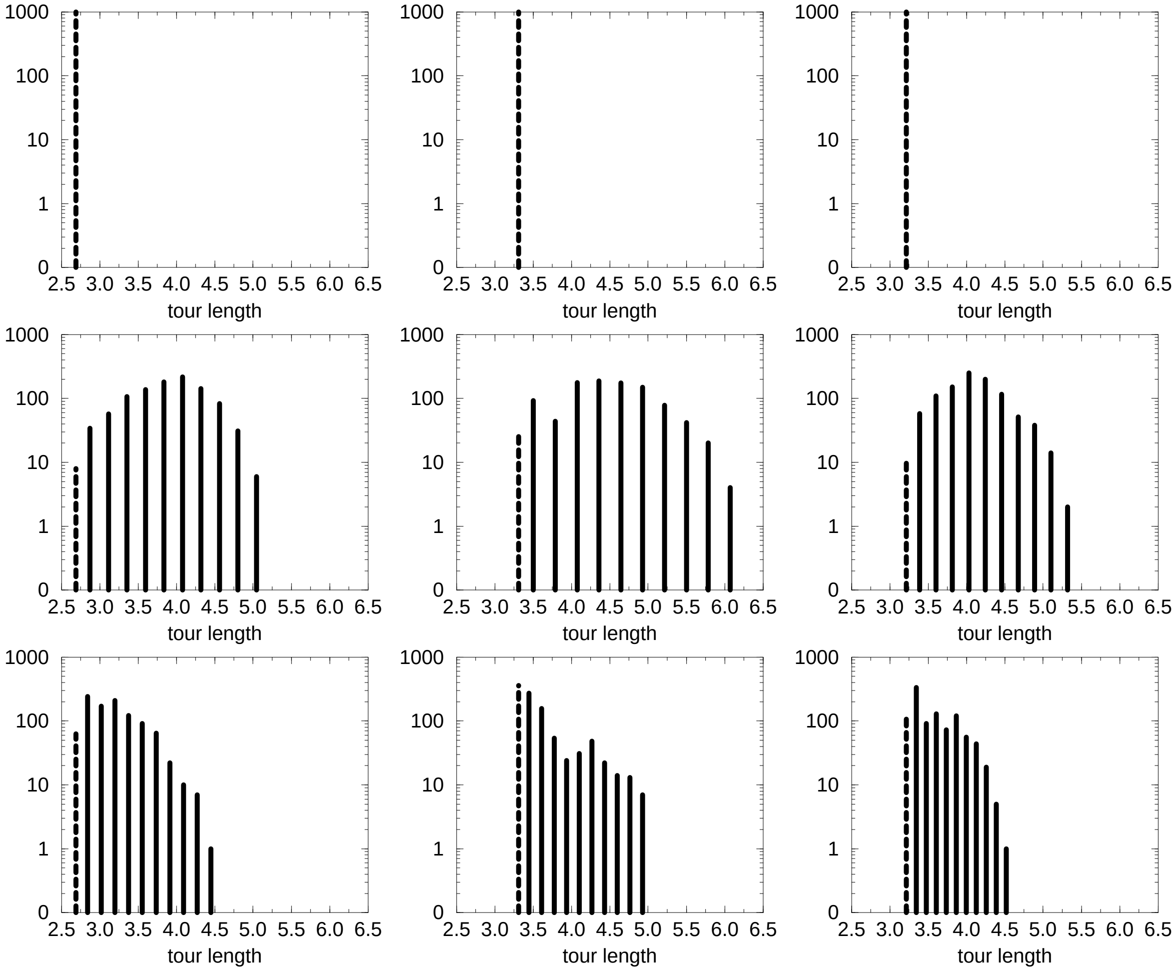

Fig. 1. Tour length histograms for 10 city problems 1, 2 and 3 (from left to right) produced by the ALH (top), the OB (middle) and the IB (bottom) methods. The dashed bars at the leftmost side indicate the numbers of times that the true minimal tour was found.

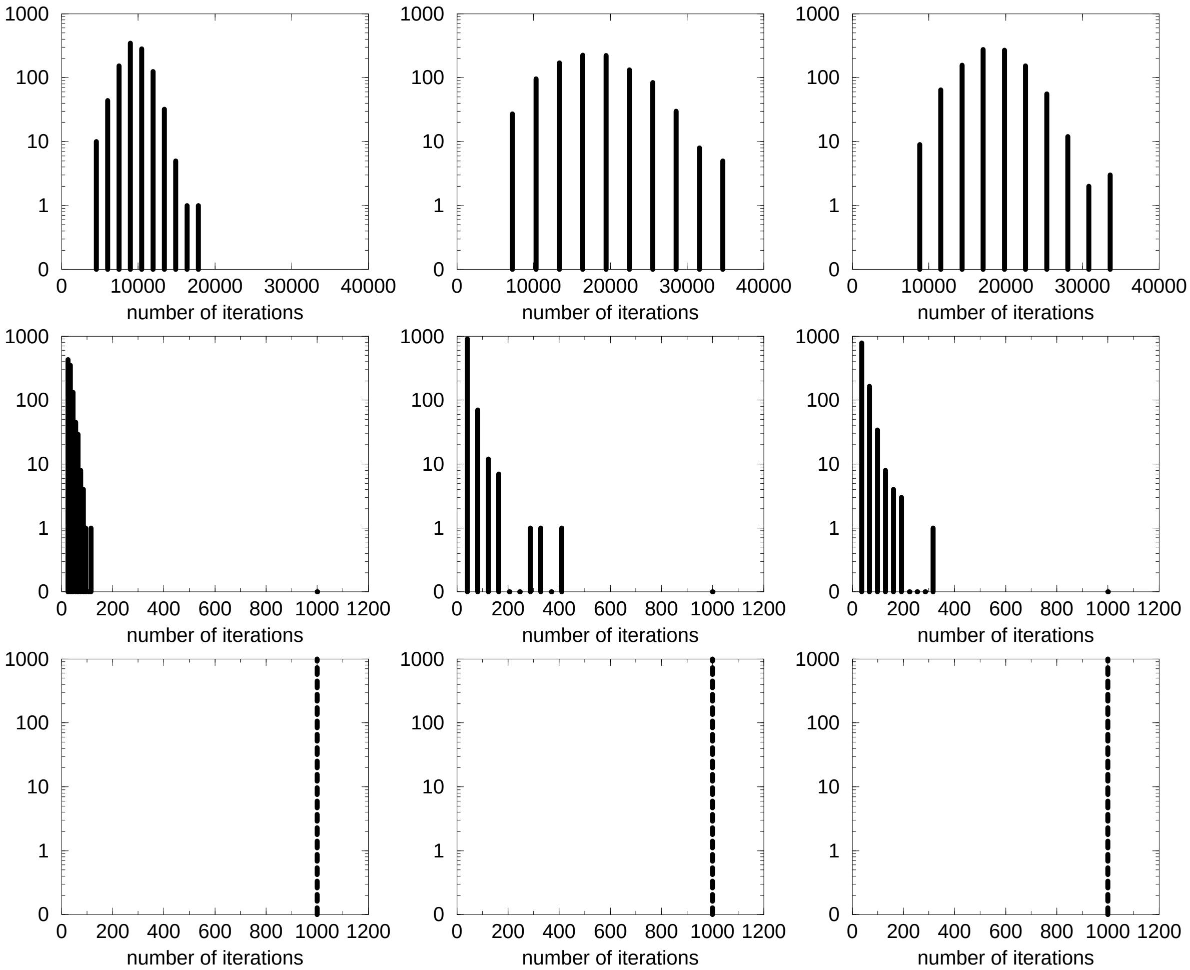

Fig. 2. Iteration number histograms for 10 city problems 1, 2 and 3 (from left to right) produced by the ALH (top), the OB (middle), and the IB (bottom) methods. The dashed bars at the rightmost side in these plots indicate the numbers of time-outs.

TABLE II

Comparison of Performance for the 10-city problems

Method	Percentage of Minimal Tours	Percentage of Valid Tours	Convergence to Feasible & Unambiguous State	Convergence Rate
ALH	100% , 100% , 100%	100% , 100% , 100%	100%	slow
OB	0.8% , 3.1% , 1.2%	100% , 99.9% , 100%	100%	fast
IB	6.3% , 35.8% , 13%	73% , 75% , 84%	0%	all time-out

TABLE III

Comparison of Performance for the 16-,22- and 48-city problems

Method	Percentage of Minimal Tours	Percentage of Valid Tours	Convergence to Feasible & Unambiguous State	Convergence Rate
ALH	0	100% , 100% , 100%	100%	slow
OB	0	100% , 99.9% , 96.4%	100%	fast
IB	0	98.9% , 90.1% , 20.3%	0%, 0%, 19.8%	mostly time-out

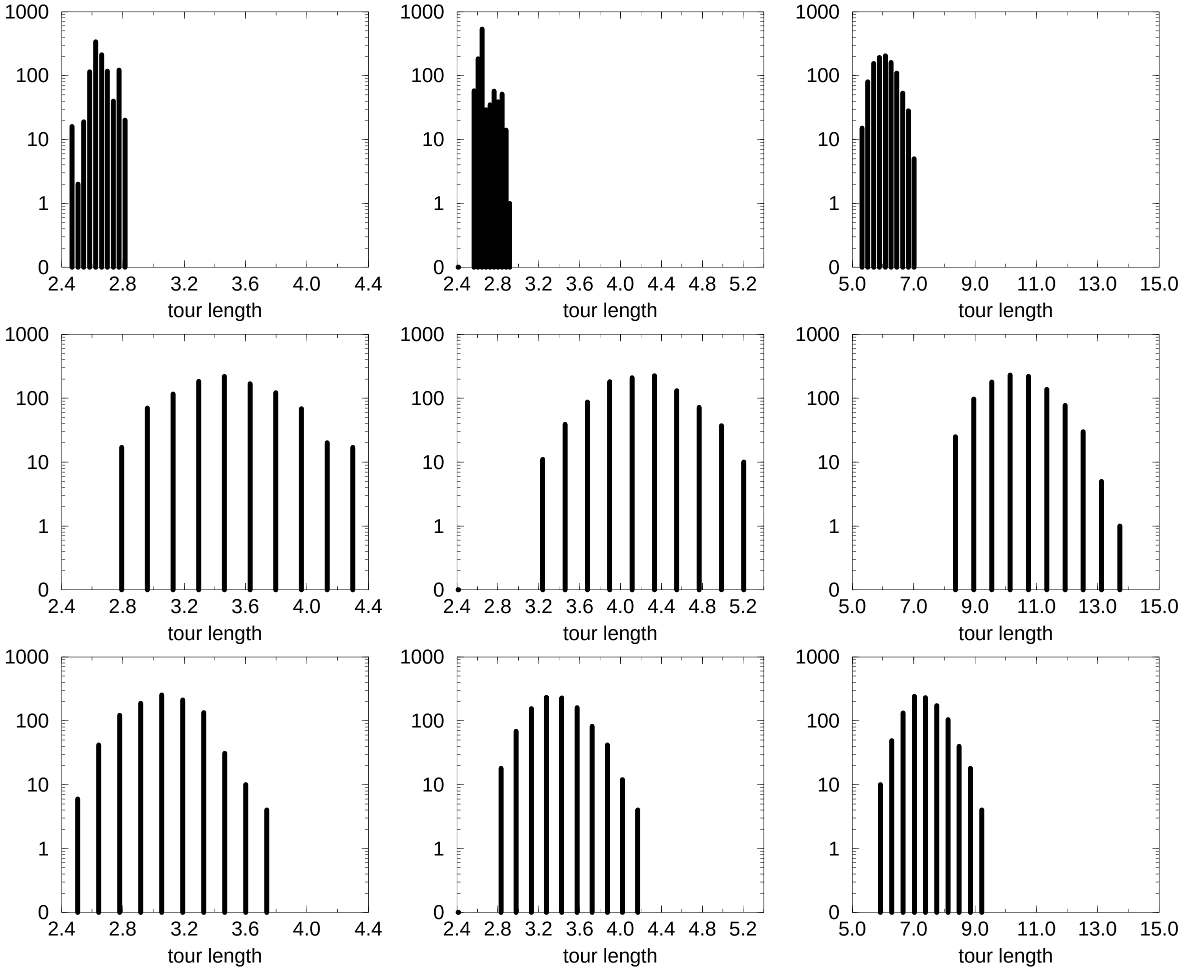

Fig. 3. Tour length histograms for 16-, 22-, and 22-city problems (from left to right) produced by the ALH (top), the OB (middle) and the IB (bottom) methods.

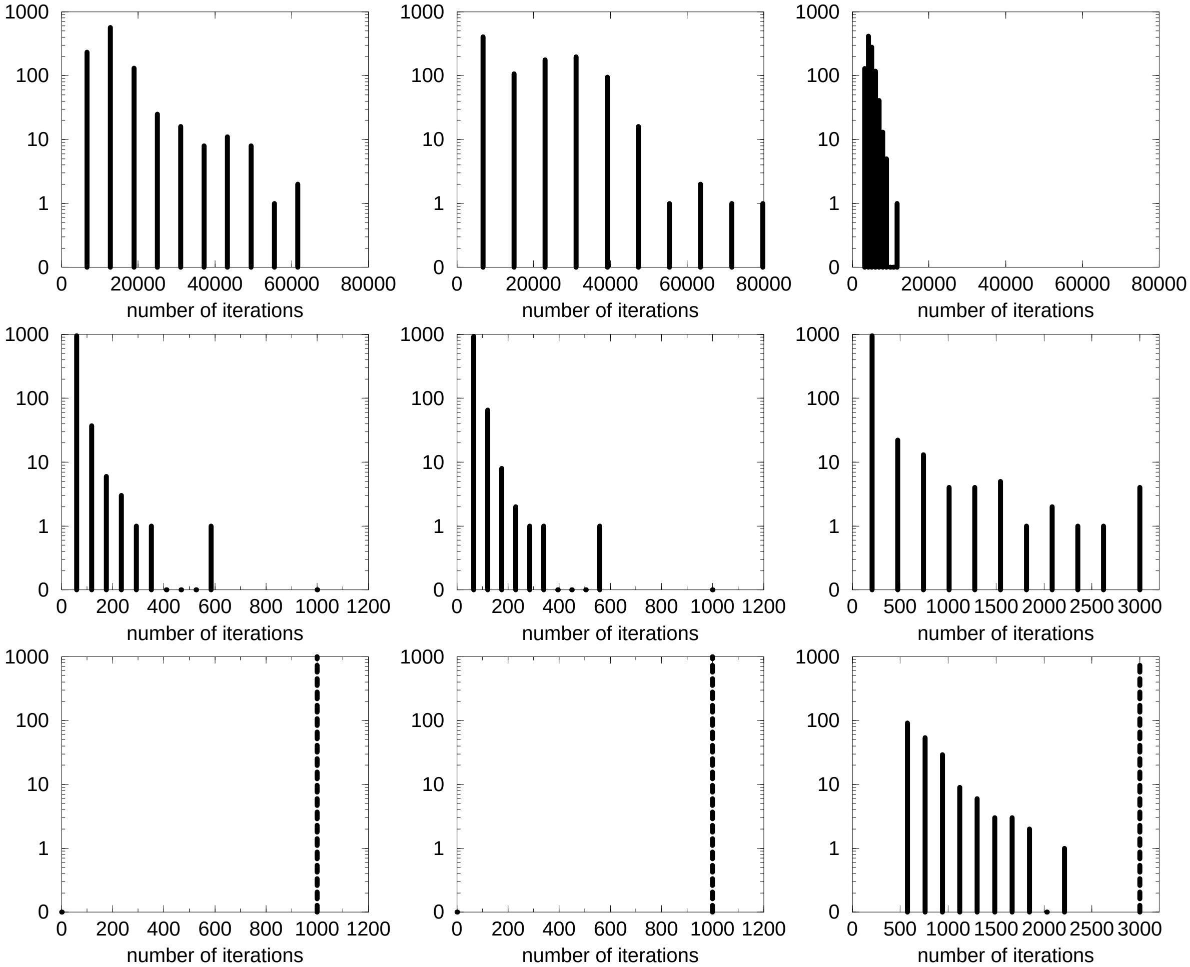

Fig. 4. Iteration number histograms for 16-, 22-, and 48-city problems (from left to right) produced by the ALH (top), the OB (middle), and the IB (bottom) methods. The dashed bars at the rightmost side indicate the numbers of time-outs.