Message-Passing for the Traveling Salesman Problem

Chang-Dong Wang Jian-Huang Lai Wei-Shi Zheng
School of Information Science and Technology, Sun Yat-sen University, Guangzhou, P. R. China.
changdongwang@hotmail.com, stsljh@mail.sysu.edu.cn, wszheng@ieee.org

Abstract

This paper exploits the graphical model with the max-sum belief propagation to solve the traveling salesman problem which is commonly solved by heuristic algorithms. Based on the visiting-ship between each city and step, we represent the optimal tour search problem by a factor graph and utilize the max-sum belief propagation algorithm to achieve the neighborhood optimal solution. By applying some mathematical tricks to simplify the original messages, we obtain an efficient message-passing algorithm.

1. Introduction

The traveling salesman problem (TSP) is a classic combinatorial optimization problem which is typically solved by the heuristic methods [7, 12, 10]. The main disadvantages of these methods are that they are not theoretically sound and strongly dependent on the initialization. To tackle this problem, this paper proposes to use the graphical model with the max-sum belief propagation to efficiently achieve the neighborhood optimal solution. Graphical model is a well-studied tool to represent structured probability distributions over large sets of random variables, and has been widely used in many applications such as data clustering [1], computer vision [5], and graph matching [9]. There are several algorithms for the exact or approximate inference in various types of graphical models [5, 3, 8, 6].

2. Graphical model and the algorithm

2.1. The factor graph model

Given a similarity matrix $[s(i,j)]_{(N+1)\times(N+1)}$ of $N+1$ cities with $s(i,j)$ denoting the similarity between city i and j (e.g. the maximum Euclidean distance between these cities *minus* the Euclidean distance between city i and j), the goal of TSP is to search for a tour (Hamiltonian circuit) of the $N+1$ cities with the maximum sum of similarities. Since the sum of similarities is independent of the ending city, without loss of generality, we fix the ending city to be the $N+1$-th city. Let $\mathbf{c} = [c_1, \dots, c_N]$ be a vector with c_t denoting the id of the city visited in step t, the goal is to search for a visiting vector such that the sum of similarities $\sum_{t=1}^{N-1} s(c_t, c_{t+1}) + s(c_N, N+1) + s(N+1, c_1)$ is maximized under the constraint $c_t \in \{1, \dots, N\}, c_t \neq c_{t'}, \forall t \neq t'$.

We adopt the binary variable model [2] to construct the factor graph based representation. Let $B = [b_{it}]_{N\times N}$ be a binary matrix denoting the visiting-ship between the first N cities and steps, such that

$$b_{it} = \begin{cases} 1 & \text{city } i \text{ is visited in step } t \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The constraint that $c_t \in \{1, \dots, N\}, c_t \neq c_{t'}, \forall t \neq t'$ is converted into *city constraint* and *step constraint* as follows. *City constraint:* Each city i must be visited in exactly one step, i.e.,

$$I_i(b_{i1}, \dots, b_{iN}) = \begin{cases} 0 & \text{if } \sum_{t=1}^{N} b_{it} = 1 \\ -\infty & \text{otherwise.} \end{cases} \tag{2}$$

Step constraint: Each step t must visit exactly one city, i.e.,

$$E_t(b_{1t}, \dots, b_{Nt}, c_t) = \begin{cases} 0 & \text{if } b_{c_t t} = 1 \& b_{it} = 0, \forall i \neq c_t \\ -\infty & \text{otherwise.} \end{cases} \tag{3}$$

The function $S_t(c_{t+1}|c_t)$ denotes the similarity from city c_t to city c_{t+1}, i.e.,

$$S_t(c_{t+1}|c_t) = \begin{cases} s(c_t, c_{t+1}) & c_t \neq c_{t+1} \\ -\infty & c_t = c_{t+1}. \end{cases} \tag{4}$$

The objective is to maximize the following function,

$$\begin{aligned} &G(B, \mathbf{c}) \\ &= \sum_{t=1}^{N-1} S_t(c_{t+1}|c_t) + S_N(N+1|c_N) + S_{N+1}(c_1|N+1) \\ &\quad + \sum_{i=1}^{N} I_i(b_{i1}, \dots, b_{iN}) + \sum_{t=1}^{N} E_t(b_{1t}, \dots, b_{Nt}, c_t). \end{aligned} \tag{5}$$

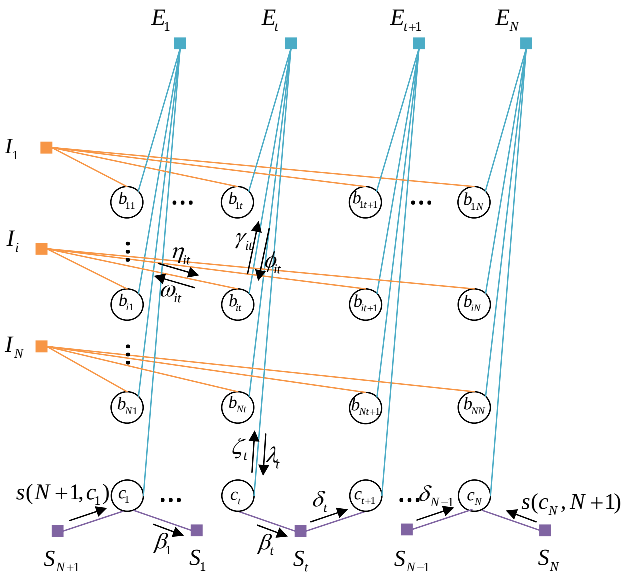

Figure 1. The factor graph based representation and BP messages. The variable nodes are depicted as usual by circles, and the function nodes are depicted by small squares. The BP messages are plotted by black arrows.

This objective function is already factorized (in the log domain) into $3N + 1$ functions. Figure 1 illustrates the factor graph based representation of (5).

2.2. Messages

To exactly search for the optimal solution for (5) is NP-hard. Many approaches have been developed to efficiently find good approximate solutions, such as [5, 3, 8, 6]. Among them, for arbitrary objective functions (hence arbitrary graphical models), message passing algorithms such as max-product belief propagation are the most popular ones [5, 4]. The max-sum (the log-domain max-product) belief propagation is a local-message-passing algorithm guaranteed to converge to the neighborhood maximum [11]. In the max-sum algorithm, the message update involves either a message from a variable to each adjacent function or that from a function to each adjacent variable. The message from a variable to a function sums together the messages from all adjacent functions except the one receiving the message [4],

$$\mu_{x\to f}(x) \leftarrow \sum_{h\in \mathrm{ne}(x)\setminus\{f\}} \mu_{h\to x}(x) \tag{6}$$

where $\mathrm{ne}(x)$ denotes the set of adjacent functions of variable x. The message from a function to a variable involves a maximization over all arguments of the function except the variable receiving the message [4],

$$\mu_{f\to x}(x) \leftarrow \max_{X\setminus\{x\}} \left[f(X) + \sum_{y\in X\setminus\{x\}} \mu_{y\to f}(y) \right] \tag{7}$$

where $X = \mathrm{ne}(f)$ is the set of arguments of the function f.

There are 8 types of belief propagation (BP) messages passing between variable nodes and function nodes as shown as the black arrow in Figure 1.

Message η_{it} : $\mu_{I_i\to b_{it}}(m), m = 0, 1.$

$$\begin{aligned}\eta_{it}(m) &= \max_{b_{it'}:t'\neq t} \left[I_i(b_{i1}, \ldots, b_{iN}) + \sum_{t':t'\neq t} \omega_{it'}(b_{it'}) \right] \\ &= \begin{cases} \sum_{t':t'\neq t} \omega_{it'}(0) & m = 1 \\ \max_{t':t'\neq t} \left[\omega_{it'}(1) + \sum_{t'':t''\notin\{t,t'\}} \omega_{it''}(0) \right] & m = 0. \end{cases}\end{aligned} \tag{8}$$

Message ω_{it} : $\mu_{b_{it}\to I_i}(m), m = 0, 1.$

$$\omega_{it}(m) = \phi_{it}(m). \tag{9}$$

Message ϕ_{it} : $\mu_{E_t\to b_{it}}(m), m = 0, 1.$

$$\begin{aligned}&\phi_{it}(m) \\ &= \max_{b_{i't}:i'\neq i, c_t} \left[E_t(b_{1t} \ldots, b_{Nt}, c_t) + \sum_{i':i'\neq i} \gamma_{i't}(b_{i't}) + \zeta_t(c_t) \right] \\ &= \begin{cases} \sum_{i':i'\neq i} \gamma_{i't}(0) + \zeta_t(i) & m = 1 \\ \max_{i':i'\neq i} \left[\gamma_{i't}(1) + \zeta_t(i') + \sum_{i'':i''\notin\{i,i'\}} \gamma_{i''t}(0) \right] & m = 0. \end{cases}\end{aligned} \tag{10}$$

Message γ_{it} : $\mu_{b_{it}\to E_t}(m), m = 0, 1.$

$$\gamma_{it}(m) = \eta_{it}(m). \tag{11}$$

Message $\beta_t, t = 1, \ldots, N-1$: $\mu_{c_t\to S_t}(m), m = 1, \ldots, N.$

$$\beta_t(m) = \begin{cases} \lambda_t(m) + s(N+1, m) & t = 1, \\ \lambda_t(m) + \delta_{t-1}(m) & t = 2, \ldots, N-1. \end{cases} \tag{12}$$

Message $\delta_t, t = 1, \ldots, N-1$: $\mu_{S_t\to c_{t+1}}(m), m = 1, \ldots, N.$

$$\begin{aligned}\delta_t(m) &= \max_{c_t} \left[S_t(c_t, m) + \beta_t(c_t) \right] \\ &= \max_{c_t\neq m} \left[s(c_t, m) + \beta_t(c_t) \right].\end{aligned} \tag{13}$$

Message λ_t : $\mu_{E_t\to c_t}(m), m = 1, \ldots, N.$

$$\begin{aligned}\lambda_t(m) &= \max_{b_{it}:i=1,\ldots,N} \left[E_t(b_{1t}, \ldots, b_{Nt}, m) + \sum_{i=1}^{N} \gamma_{it}(b_{it}) \right] \\ &= \gamma_{mt}(1) + \sum_{i:i\neq m} \gamma_{it}(0).\end{aligned} \tag{14}$$

Message $\zeta_t : \mu_{c_t \to E_t}(m), m = 1, \ldots, N$.

$$\zeta_t(m) = \begin{cases} s(N+1, m) & t = 1, \\ \delta_{t-1}(m) & t = 2, \ldots, N-1, \\ \delta_{N-1}(m) + s(m, N+1) & t = N. \end{cases} \tag{15}$$

2.3. Message simplification

Like in the affinity propagation algorithm [1, 2], we can simplify the above messages by applying some mathematical tricks. First of all, for each message associated with the binary variable, we introduce the difference between those with argument 1 and 0. That is, $\tilde{\eta}_{it} = \eta_{it}(1) - \eta_{it}(0)$, $\tilde{\omega}_{it} = \omega_{it}(1) - \omega_{it}(0)$, $\tilde{\phi}_{it} = \phi_{it}(1) - \phi_{it}(0)$, $\tilde{\gamma}_{it} = \gamma_{it}(1) - \gamma_{it}(0)$. For the remaining messages, they are viewed as the sum of variable (with respect to the corresponding argument) and constant. That is, $\beta_t(m) = \tilde{\beta}_t(m) + \bar{\beta}_t$, $\delta_t(m) = \tilde{\delta}_t(m) + \bar{\delta}_t$, $\lambda_t(m) = \tilde{\lambda}_t(m) + \bar{\lambda}_t$, and $\zeta_t(m) = \tilde{\zeta}_t(m) + \bar{\zeta}_t$.

Message $\tilde{\eta}_{it}$:

$$\begin{aligned} \tilde{\eta}_{it} =& \eta_{it}(1) - \eta_{it}(0) \\ =& \sum_{t':t' \neq t} \omega_{it'}(0) - \max_{t':t' \neq t} \left[\omega_{it'}(1) + \sum_{t'':t'' \notin \{t,t'\}} \omega_{it''}(0) \right] \\ =& - \max_{t':t' \neq t} \left[\omega_{it'}(1) - \omega_{it'}(0) \right] = - \max_{t':t' \neq t} \left[\tilde{\omega}_{it'} \right]. \end{aligned} \tag{16}$$

Message $\tilde{\omega}_{it}$:

$$\tilde{\omega}_{it} = \omega_{it}(1) - \omega_{it}(0) = \phi_{it}(1) - \phi_{it}(0) = \tilde{\phi}_{it}. \tag{17}$$

Message $\tilde{\phi}_{it}$:

$$\begin{aligned} \tilde{\phi}_{it} &= \phi_{it}(1) - \phi_{it}(0) \\ &= \left(\sum_{i':i' \neq i} \gamma_{i't}(0) + \zeta_t(i) \right) \\ &\qquad - \max_{i':i' \neq i} \left[\gamma_{i't}(1) + \zeta_t(i') + \sum_{i'':i'' \notin \{i,i'\}} \gamma_{i''t}(0) \right] \\ &= - \max_{i':i' \neq i} \left[(\gamma_{i't}(1) - \gamma_{i't}(0)) + \zeta_t(i') - \zeta_t(i) \right] \\ &= - \max_{i':i' \neq i} \left[\tilde{\gamma}_{i't} + \tilde{\zeta}_t(i') \right] + \tilde{\zeta}_t(i). \end{aligned} \tag{18}$$

Message $\tilde{\gamma}_{it}$:

$$\tilde{\gamma}_{it} = \gamma_{it}(1) - \gamma_{it}(0) = \eta_{it}(1) - \eta_{it}(0) = \tilde{\eta}_{it}. \tag{19}$$

Message $\beta_t(m)$:

$$\beta_t(m) = \begin{cases} \tilde{\lambda}_t(m) + \bar{\lambda}_t + s(N+1, m) & t = 1, \\ \tilde{\lambda}_t(m) + \bar{\lambda}_t + \tilde{\delta}_{t-1}(m) + \bar{\delta}_{t-1} & t = 2, \ldots, N-1. \end{cases} \tag{20}$$

Message $\delta_t(m)$:

$$\delta_t(m) = \max_{c_t \neq m} \left[s(c_t, m) + \tilde{\beta}_t(c_t) \right] + \bar{\beta}_t, \forall t = 1, \ldots, N-1. \tag{21}$$

Message $\lambda_t(m)$:

$$\lambda_t(m) = (\tilde{\gamma}_{mt} + \gamma_{mt}(0)) + \sum_{i:i \neq m} \gamma_{it}(0) = \tilde{\gamma}_{mt} + \sum_{i=1}^{N} \gamma_{it}(0). \tag{22}$$

Message $\zeta_t(m)$:

$$\begin{aligned} &\zeta_t(m) \\ &= \begin{cases} s(N+1, m) & t = 1, \\ \tilde{\delta}_{t-1}(m) + \bar{\delta}_{t-1} & t = 2, \ldots, N-1, \\ \tilde{\delta}_{N-1}(m) + \bar{\delta}_{N-1} + s(m, N+1) & t = N. \end{cases} \end{aligned} \tag{23}$$

2.4. Message summary

According to the trick used in the binary model of AP [2], for each message with the binary argument, we only consider and update its difference denoted using "$\tilde{}$". And similar to [1], for each message with non-binary argument, the variable part denoted using "$\tilde{}$" is considered and updated. Consequently, we have the following simplified messages.

$\forall i, t = 1, \ldots, N$:

$$\tilde{\phi}_{it} = - \max_{i':i' \neq i} \left[\tilde{\gamma}_{i't} + \tilde{\zeta}_t(i') \right] + \tilde{\zeta}_t(i), \tag{24}$$

$$\tilde{\gamma}_{it} = - \max_{t':t' \neq t} \left[\tilde{\phi}_{it'} \right]. \tag{25}$$

$\forall m = 1, \ldots, N,\ \forall t = 1, \ldots, N-1$:

$$\beta_t(m) = \begin{cases} \tilde{\lambda}_1(m) + s(N+1, m) & t = 1, \\ \tilde{\lambda}_t(m) + \tilde{\delta}_{t-1}(m) & t = 2, \ldots, N-1, \end{cases} \tag{26}$$

$$\delta_t(m) = \max_{n \neq m} \left[s(n, m) + \tilde{\beta}_t(n) \right]. \tag{27}$$

$\forall m, t = 1, \ldots, N$:

$$\lambda_t(m) = \tilde{\gamma}_{mt}, \tag{28}$$

$$\zeta_t(m) = \begin{cases} s(N+1, m) & t = 1, \\ \tilde{\delta}_{t-1}(m) & t = 2, \ldots, N-1, \\ \tilde{\delta}_{N-1}(m) + s(m, N+1) & t = N. \end{cases} \tag{29}$$

In the above simplified messages, the message pair of $\tilde{\eta}_{it}$ and $\tilde{\omega}_{it}$ is eliminated by directly replacing these two messages with $- \max_{t':t' \neq t} \left[\tilde{\phi}_{it'} \right]$ and $\tilde{\phi}_{it}$ respectively.

2.5. Estimate $\mathbf{c}$

To estimate the value of an element c_t, we sum together all incoming messages to c_t and take the value $\hat{c}_t$ that maximizes the sum. That is,

$$\begin{aligned}\hat{c}_t &= \arg\max_{c_t=1,\ldots,N} [\lambda_t(c_t) + \delta_{t-1}(c_t)] \\ &= \arg\max_{c_t=1,\ldots,N} \left[\tilde{\lambda}_t(c_t) + \tilde{\delta}_{t-1}(c_t)\right], \forall t = 2, \ldots, N-1,\end{aligned} \tag{30}$$

$$\begin{aligned}\hat{c}_1 &= \arg\max_{c_1=1,\ldots,N} [\lambda_1(c_1) + s(N+1, c_1)] \\ &= \arg\max_{c_1=1,\ldots,N} \left[\tilde{\lambda}_1(c_1) + s(N+1, c_1)\right],\end{aligned} \tag{31}$$

$$\begin{aligned}\hat{c}_N &= \arg\max_{c_N=1,\ldots,N} [\delta_{N-1}(c_N) + \lambda_N(c_N) + s(c_N, N+1)] \\ &= \arg\max_{c_N=1,\ldots,N} \left[\tilde{\delta}_{N-1}(c_N) + \tilde{\lambda}_N(c_N) + s(c_N, N+1)\right].\end{aligned} \tag{32}$$

2.6. The algorithm

The overall algorithm of the proposed approach is summarized as follows. Given the similarity matrix s and the dampening factor θ (e.g. 0.5), we first initialize all messages $\tilde{\phi}_{it}$, $\tilde{\gamma}_{it}$, $\beta_t(m)$, $\delta_t(m)$, $\lambda_t(m)$ as zeros. Then each type of these messages are sequentially updated via equations (24) to (29) with additional dampening, i.e. $\mu \leftarrow \theta\mu^{\text{old}} + (1-\theta)\mu^{\text{new}}$. The message-updating procedure may be terminated after the local decisions (i.e. the estimate of $\mathbf{c}$) stay constant for some number of iterations t_{conv}, or after a fixed number of iterations t_{max}.

3. Empirical results

This section reports some empirical results with the dampening factor θ set to 0.5, $t_{conv} = 5$, and $t_{max} = 1000$.

We first demonstrate the result on a small city group consisting of 5 cities with the similarity matrix given as follows

$$s = \begin{pmatrix} 0.8 & 10.1 & 12.5 & 0.1 & 0.6 \\ 0.9 & 0.2 & 0.9 & 0.4 & 0.1 \\ 0.1 & 0.5 & 0.9 & 0.9 & 0.8 \\ 0.9 & 0.9 & 0.5 & 0.8 & 0.9 \\ 0.6 & 0.009 & 1.8 & 0.9 & 0.6 \end{pmatrix}. \tag{33}$$

The tour produced by the proposed approach is

$$5 \to 2 \to 4 \to 3 \to 1 \to 5 \tag{34}$$

with the tour length being 1.609, which is equal to the overall minimum tour length. Our approach only requires 10 iterations to reach this good result; on the other hand the number of all possible permutations of these 5 cities is 120 which implies that we have to try 120 times for the optimal solutions using exhaustive search.

On the other synthetic testing set of 100 cities, only about 1000 iterations are required to reach the optimal solution. Compared to the typically used heuristic algorithms, the main advantages of the proposed algorithm are that, it is based on the sound theory of the probabilistic graphical models with the max-sum belief propagation, and it is deterministic without random initialization. Additionally, it is quite efficient.

4. Conclusions

This paper presents the use of the message-passing algorithm to solve the traveling salesman problem. A new graphical model based representation is proposed and the max-sum belief propagation is utilized to generate the neighborhood optimal solution. In the future work, we will try the Junction-Tree algorithm for an exact inference in the derived graphical model.

References

[1] B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315:972–976, 2007. 1, 3

[2] I. E. Givoni and B. J. Frey. A binary variable model for affinity propagation. *Neural Computation*, 21(6):1589–1600, June 2009. 1, 3

[3] S. Gould, F. Amat, and D. Koller. Alphabet SOUP: A framework for approximate energy minimization. In *CVPR*, 2009. 1, 2

[4] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, 47(2):498–519, Feb. 2001. 2

[5] M. P. Kumar and P. Torr. Fast memory-efficient generalized belief propagation. In *ECCV*, 2006. 1, 2

[6] Y. Lim, K. Jung, and P. Kohli. Energy minimization under constraints on label counts. In *ECCV*, 2010. 1, 2

[7] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21:498–516, 1973. 1

[8] J. McAuley and T. Caetano. Exploiting within-clique factorizations in Junction-Tree algorithms. In *NIPS Workshop*, 2009. 1, 2

[9] J. McAuley, T. Caetano, and M. S. Barbosa. Graph rigidity, cyclic belief propagation, and point pattern matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30:2047–2054, 2008. 1

[10] T. Öncan, I. K. Altınel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36:637–654, 2009. 1

[11] Y. Weiss and W. T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Trans. Inf. Theory*, 47(2):736–744, Feb. 2001. 2

[12] X.-F. Xie and J. Liu. Multiagent optimization system for solving the traveling salesman problem (TSP). *IEEE Trans. Syst., Man, Cybern. Part B, Cybern.*, 39(2):489–502, April 2009. 1