Iterative Message Passing Algorithm for Vertex-disjoint Shortest Paths

Guowei Dai, Longkun Guo, Gregory Gutin, Xiaoyan Zhang*, Zan-Bo Zhang

Abstract

As an algorithmic framework, message passing is extremely powerful and has wide applications in the context of different disciplines including communications, coding theory, statistics, signal processing, artificial intelligence and combinatorial optimization. In this paper, we investigate the performance of a message-passing algorithm called min-sum belief propagation (BP) for the vertex-disjoint shortest k-path problem (k-VDSP), and derive the iterative message-passing update rules. As the main result of this paper, we prove that for a weighted digraph G of order n, BP algorithm converges to the unique optimal solution of k-VDSP on G within $O(n^2 w_{max})$ iterations, provided that the weight w_e is nonnegative integral for each arc $e \in E(G)$, where $w_{max} = \max\{w_e : e \in E(G)\}$. To the best of our knowledge, this is the first instance where BP algorithm is proved correct for NP-hard problems. Additionally, we establish the extensions of k-VDSP to the versions of multiple sources or sinks.

Key words: Belief propagation, Message-passing algorithm, Vertex-disjoint shortest path

I. Introduction

Belief propagation (BP) is a distributed, message-passing heuristic algorithm for solving optimization and inference problems on various graphical models. Since the proposition of BP algorithm by Pearl in 1988 [20], the message-passing algorithm based on BP has shown its power as an algorithmic framework and has wide applications in the context of variety of disciplines including satisfiability in discrete optimization [1], [8], [18], [19], error correcting code in information theory [12], [14], [17], [21], and data clustering in machine learning [9]. BP algorithm is known as essentially an approximation of the dynamic programming when the underlying graph has no cycles [12], [20], [25]. Specifically, BP algorithm provides a natural parallel iterative version of the dynamic programming in which variable vertices pass messages between each other along arcs on graphical models. Surprisingly, even for graphs with many cycles, the BP algorithm performs well in practice and has empirically been shown to give good results [18], [21]. While BP algorithms have been shown empirically to be effective in solving many instances of optimization problems, the theoretical analysis of the performance of BP algorithm remains far from complete.

Some progress has been made in understanding their convergence and accuracy of BP algorithms for several optimization and inference problems, see, e.g., [3]–[7], [13], [22], [23]. As a major breakthrough, Bayati et al. [4] and Cheng et al. [5] independently simplified the BP algorithm to obtain two essentially same algorithms for the maximum weight matching (MWM) on a bipartite graph. They established the convergence of the BP

G. Dai is with School of Mathematical Science & Institute of Mathematics, Nanjing Normal University, Nanjing 210023, China (e-mail: guoweidai@njnu.edu.cn).

L. Guo is with Department of Computer Science, Qilu University of Technology, Jinan, China (e-mail: longkun.guo@gmail.com).

G. Gutin is with Department of Computer Science, Royal Holloway University of London, Egham, UK (e-mail: gutin@cs.rhul.ac.uk).

X. Zhang is with School of Mathematical Science & Institute of Mathematics, Nanjing Normal University, Nanjing, China (e-mail: royxyzhang@gmail.com, zhang_njnu@aliyun.com).

Z. Zhang is with School of Statistics & Mathematics, and Institute of Artificial Intelligence & Deep Learning, Guangdong University of Finance & Economics, Guangzhou, China (e-mail: zanbozhang@gdufe.edu.cn).

algorithm for MWM, provided that the optimal solution is unique. Bayati et al. [3] as well as Sanghavi et al. [22] generalized the result by showing the convergence of BP algorithm for the min-cost b-matching problem on arbitrary graphs, provided that the corresponding linear programming (LP) relaxation has a unique integral optimal solution. Note that the weighted matching problem on bipartite graphs can be viewed as a special case of the minimum cost flow (MCF) problem. Gamarnik et al. [13] proved that BP algorithm for MCF converges to the optimal solution if its optimal solution is unique. Recently, Even and Halabi [7] developed a BP algorithm for the covering and packing problem and established that BP algorithm converges to the optimal solution if its LP relaxation has a unique integral optimal solution. Sanghavi et al. [23] investigated the performance of BP algorithm for the max-weight independent set problem and established a one-sided relation between BP algorithm and its LP relaxation. Furthermore, an example in [23] shows that BP algorithm is unlikely to solve the general linear programming problem.

Graph routing problems have already attracted intensive research from mathematicians and computer scientists starting from early 1970s. One of the most well-known graph routing problems is the travelling salesman problem (TSP), for which Gutin and Punnen [10] provided a compendium of results. In particular, Chapter 6 of [10] describes a somewhat unexpected result that for any number n of vertices there is an infinite number of TSP instances (both asymmetric and symmetric) such that the greedy algorithm outputs the unique worst possible solution. The same result holds for the TSP nearest neighbor algorithm. These results were proved in [11] and the TSP greedy algorithm result was generalized to other combinatorial optimization problems in [2].

As is a class of graph routing problems, the vertex-disjoint shortest k-path problem (k-VDSP) was first introduced by Suurballe [24]. An objective of k-VDSP is to find k internally vertex-disjoint paths from given source s to sink t, with minimum total length. Note that k-VDSP is strongly NP-hard when $k \geq 2$ [15], and it will be reduced to the classic shortest s-t path problem when $k = 1$. Vertex-disjoint paths are usually used in communication networks for reliability of transmission between a given source and sink. In this paper, we focus primarily on the performance of the Min-Sum BP algorithm for finding the optimal solution of k-VDSP.

A. Our Contributions

The contributions of this paper, in detail, are as follows. First, we derive a message-passing algorithm based on BP for finding the optimal solution of k-VDSP. Then we establish that for any weighted digraph G with n vertices, as long as the optimal solution is unique, our algorithm converges to the optimal solution x^* within $(\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n$ iterations, where U and $o(x^*)$ are the maximum weight of a simple directed path and minimum weight of a directed cycle in the residual network G_{x^*}, respectively. Note that we develop new and more complex rules in our proof since the constraints of k-VDSP are more complex than those of the previous problems in [4], [7], [13], [22], [23]. Next, we show that the Min-Sum BP algorithm converges to the unique optimal solution in $O(n^2 w_{max})$ iterations, provided that the weight w_e is nonnegative integral for each arc $e \in E(G)$, where $w_{max} = \max\{w_e : e \in E(G)\}$. Additionally, we extend our analysis to establish the extensions of k-VDSP to the versions of multiple sources or sinks.

It is known that BP algorithm is unlikely to solve the general linear programming problem by means of a counterexample [23]. Thus, our results extend the scope of the problems that are provably solvable by the BP algorithm. To the best of our knowledge, this is the first instance where BP algorithm is proved correct for NP-hard problems. We believe that our methods can help to analyse the convergence and accuracy of BP algorithms for other NP-hard problems with more complex constraints.

II. Preliminaries

A. Problem Statement

The input to the vertex-disjoint shortest k-path problem (k-VDSP) is a weighted digraph $G = (V(G), E(G), w)$, where $V(G), E(G)$ denote the set of vertices and arcs (i.e., directed edges) in G, respectively, and $w : E \to \mathbb{R}^+$ is a weight function. The weight $w(P)$ of a path P is defined as the sum of the weights of its arcs. Several paths are said to be *internally vertex-disjoint* if for any two paths of them, there exits no vertices in common except at the terminals. For a given weighted digraph G with source $s \in V(G)$ and sink $t \in V(G)$, the problem k-VDSP aims to find k internally vertex-disjoint paths from s to t, denoted by $P_1, P_2, ..., P_k$, such that $\sum_{i=1}^{k} w(P_i)$ is minimized. Let $\mathcal{P} = \{P_1, P_2, ..., P_k\}$ and $E(\mathcal{P}) = \cup_{i=1}^{k} E(P_i)$, where $E(P_i)$ denotes the set of arcs in P_i. For each $e \in E(G)$, define x_e as an indicator variable that $x_e = 1$ if $e \in E(\mathcal{P})$, and $x_e = 0$ else. Then those arcs belong to $X = \{e \in E(G) : x_e = 1\}$ correspond exactly to the k internally vertex-disjoint paths that $P_1, P_2, ..., P_k$ in G. So, for any k internally vertex-disjoint paths from s to t, it could be represented by $x = \{x_e : e \in E(G)\}$ where x_e is defined the same as before.

We use w_e to denote the weight on e for any arc $e \in E(G)$. For any vertex $i \in V(G)$, denote the sets of out-neighbors and in-neighbors of i in G by $N_i^+ = \{j : ij \in E(G)\}$ and $N_i^- = \{j : ji \in E(G)\}$, respectively, and let $N_i = N_i^+ \cup N_i^-$. Throughout the paper, we assume there exist no in-neighbors of source vertex and out-neighbors of sink vertex, that is, $N_s^- = N_t^+ = \emptyset$. Let x_e be the 0-1 value assigned to each arc $e \in E(G)$. Then the k-VDSP on graph $G = (V(G), E(G), w)$ can also be formulated as the follows:

$$\min \quad \sum_{e \in E(G)} w_e x_e \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in N_s^+} x_{sj} = \sum_{j \in N_t^-} x_{jt} = k, \tag{2}$$

$$\sum_{j \in N_i^+} x_{ij} - \sum_{j \in N_i^-} x_{ji} = 0, \; \forall\, i \in V(G) \setminus \{s,t\}; \tag{3}$$

$$\sum_{j \in N_i} x_{ij} \in \{0, 2\}, \qquad \forall\, i \in V(G) \setminus \{s,t\}; \tag{4}$$

$$x_e \in \{0, 1\}, \qquad \forall\, e \in E(G). \tag{5}$$

The type of constraints (2) and (3) state that there are exactly k paths from s to t. The third type of constraints (4) state that these k paths are internally vertex-disjoint. Note that on the premise that (3) and (5) are satisfied, the type of constraints (4) hold if and only if

$$\sum_{j \in N_i^+} x_{ij} + \sum_{j \in N_i^-} x_{ji} \leq 2, \; \forall\, i \in V(G) \setminus \{s,t\}.$$

Define a vertex demand function $f : V(G) \to \mathbb{Z}$ that $f_s = k, f_t = -k$ and $f_i = 0$ for any $i \in V(G) \setminus \{s,t\}$. Then the k-VDSP on G can be formulated as the following integer programming problem (*IP*):

$$\begin{aligned}
\min \quad & \sum_{e \in E} w_e x_e \\
\text{s.t.} \quad & \sum_{j \in N_i^+} x_{ij} - \sum_{j \in N_i^-} x_{ji} = f_i, && \forall\, i \in V(G); \\
& \sum_{j \in N_i^+} x_{ij} + \sum_{j \in N_i^-} x_{ji} \leq 2, && \forall\, i \in V(G) \setminus \{s,t\}; \\
& x_e \in \{0, 1\}, && \forall\, e \in E(G).
\end{aligned}$$

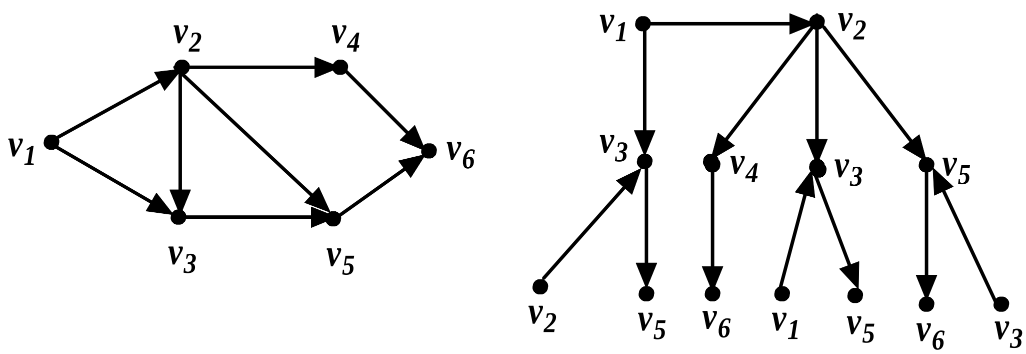

Fig. 1: An example of a 2-level computation tree $T^2_{v_1v_2}$ with root v_1v_2.

Let x be the feasible solution of the integer programming problem (*IP*) above. Then the optimal solution of k-VDSP can be defined as:

$$x^* = \arg\min_x \sum_{e\in E(G)} w_e x_e.$$

B. Computation tree

Here we introduce the concept of *rooted tree* and *computation tree*. A connected acyclic graph (i.e., contains no cycles) is called a *tree*. For any nontrivial tree, it must contain a vertex which has exactly one neighbor. Such a vertex in a tree is also called a *leaf* of the tree. Throughout of the paper, we define a rooted tree T_r as a tree T with a specified arc r, called the *root* of T. It should be noted that the definition of the root of a tree sometimes refers to a specified vertex, in contrast to a root as the root. In a tree, any two vertices are connected by exactly one path. We denote the unique path connecting vertices i and j in a tree T by iTj. For a rooted tree T_r with root r, the *level* of a vertex j in T_r is the length of the path rTj, and each vertex on the path rTj is called an *ancestor* of j. For two adjacent vertices i, j in T, if i is an ancestor of j, then i is also called a *parent* of j, and j is a *child* of i.

We use T^Q_r to denote the Q-level computation tree associated with arc r as the root. An example of computation tree can be seen in Fig. 1. Denote the set of vertices and arcs in T^Q_r by $V(T^Q_r)$ and $E(T^Q_r)$, respectively. Each vertex or arc of T^Q_r is a duplicate of some vertex or arc of the original graph G. Define the mapping $\gamma^Q_r : V(T^Q_r) \to V$ such that if $i' \in V(T^Q_r)$ is a duplicate of $i \in V(G)$, then $\gamma^Q_r(i') = i$. Denote by $L(T^Q_r)$ the set of leaves of T^Q_r. For any $i' \in V(T^Q_r)$, denote by $P(i')$ the parent of i' in T^Q_r. It is essentially the breadth-first search tree of G (with repetition of vertices allowed) starting from r up to depth Q. In detail, we inductively define T^Q_r as the following rules.

- Let $uv \in E(G)$. Then the computation tree T^0_r consists of two vertices u', v' and an arc $u'v'$, such that $\gamma^0_r(u') = u$ and $\gamma^0_r(v') = v$. The arc $r = u'v'$ is considered the "root" of T^0_r, and vertices u', v' are considered to be at 0-level of T^0_r.
- Inductively, suppose that we defined a tree T^{Q-1}_r, such that for any $i', j' \in V(T^{Q-1}_r)$, $i'j' \in E(T^{Q-1}_r)$ if and only if $\gamma^{Q-1}_r(i')\gamma^{Q-1}_r(j') \in E(G)$. The computation tree T^Q_r contains T^{Q-1}_r as a subtree, which can be obtained by adding vertices to $V(T^{Q-1}_r)$ and arcs to $E(T^{Q-1}_r)$ as follows. For each leaf vertex $i' \in L(T^{Q-1}_r)$, add node j' to expand $V(T^{Q-1}_r)$ and add arc $i'j'$ or $j'i'$ to expand $E(T^{Q-1}_r)$ if there is a vertex $j \in V(G)$ such that $ij \in E(G)$ or $ji \in E(G)$ with $\gamma^{Q-1}_r(i') = i$, and $\gamma^{Q-1}_r(P(i')) \neq j$. In this

case, define $P(j') = i'$, the map $\gamma_r^Q(j') = j$, and level of j' as Q. Indeed, γ_r^Q is identical to γ_r^{Q-1} for vertices in $V(T_r^{Q-1}) \subseteq V(T_r^Q)$.

- For any $e = ij \in E(G)$, the arc from i' to j' in T_r^Q is also denoted by e for simplicity and is assigned the same weight w_e as that in G, where $\gamma_r^Q(i') = i$ and $\gamma_r^Q(j') = j$.

In what follows, we shall drop reference to r, Q in notation of γ_r^Q when clear from context and abuse notation by denoting $\gamma(i'j') = \gamma(i')\gamma(j')$.

Now assume there is a k-VDSP problem stated for a graph $G = (V(G), E(G), w)$ with given source s and sink t. We define the induced k-VDSP problem, denoted by VDSP$_r^Q$, on computation tree T_r^Q. Given a root r, let $V^o(T_r^Q) \subset V(T_r^Q)$ denote the set of all the vertices but the leaves of T_r^Q. Then the problem VDSP$_r^Q$ can be formulated as follows:

$$\begin{aligned}
\min \quad & \sum_{e \in E(T_r^Q)} w_e y_e \\
\text{s.t.} \quad & \sum_{j' \in N_{i'}^+(T_r^Q)} y_{i'j'} - \sum_{j' \in N_{i'}^-(T_r^Q)} y_{j'i'} = f_{\gamma(i')}, && \forall\ i' \in V^o(T_r^Q); \\
& \sum_{j' \in N_{i'}^+(T_r^Q)} y_{i'j'} + \sum_{j' \in N_{i'}^-(T_r^Q)} y_{j'i'} \le 2, && \forall\ i' \in V^o(T_r^Q) \setminus \{s', t'\}; \\
& y_e \in \{0, 1\}, && \forall\ e \in E(T_r^Q),
\end{aligned}$$

where $\gamma(s') = s, \gamma(t') = t$.

Notation. The computation tree is locally equivalent to the original graph, which means one can view the iterative process of BP algorithm as sending the messages along the way from leaf vertices to the root in the computation tree. All the vertices on computation tree will send messages to their parents at each iteration, and the direction of message-passing is independent of the direction of those arcs. One can guess that the BP algorithm for VDSP$_r^Q$ works quite similar as BP algorithm for k-VDSP on the original graph, and the reasoning will be formalized in the Lemma 4.1.

III. Min-Sum BP Algorithm for k-VDSP

A. *Factorized optimization problem and factor graph*

Consider the optimization problem ($\mathscr{P}$) as following:

$$\begin{aligned}
\min \quad & \sum_{i \in V} \phi_i(x_i) + \sum_{D \in \mathscr{D}} \psi_D(x_D) \\
\text{s.t.} \quad & x_i \in \mathbb{R},\ \forall i \in V,
\end{aligned}$$

where V is a finite set of *variables* and $\mathscr{D}$ is a finite collection of subsets of V representing constraints. Here $\phi_i : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ and $\psi_D : \mathbb{R}^{|D|} \to \mathbb{R} \cup \{\infty\}, \forall D \in \mathscr{D}$ are extended real-valued functions, where each ϕ_i is called a variable function and each ψ_D is called a factor function. We also call the optimization problem ($\mathscr{P}$) a *factorized optimization problem*.

Next, we introduce the concept of a *factor graph* of a factorized optimization problem, which can be referred to [16]. on problem. A factor graph $F_{\mathscr{P}}$ of ($\mathscr{P}$) is a bipartite graph with one partition containing variables V and the other partition containing factor vertices $\mathscr{D}$ corresponding to the constraints, and there is an edge $(i, D) \in V \times D$ if and only if $i \in D$.

B. Algorithm

It is well-known that BP algorithms are always viewed as heuristic algorithms for factorized optimization problems and operate by passing messages iteratively with variables and factors. Next, we will represent k-VDSP as a factorized optimization problem. Let E_i be the set of arcs incident to i and $x_{E_i} = \{x_e : e \in E_i\}$, where $x_e \in \{0,1\}$ and x is a solution of k-VDSP. Recall that $f_s = k, f_t = -k$ and $f_i = 0$ for any $i \in V(G) \setminus \{s,t\}$. We define the factor and variable functions ϕ, ψ for each $e \in E(G), i \in V(G)$, respectively as follows: $\phi_e(x_e) = w_e x_e$ if $x_e \in \{0,1\}$, otherwise $\phi_e(x_e) = +\infty$; and

$$\psi_i(x_{E_i}) = \begin{cases} 0 & \text{if } i \in \{s,t\} \text{ and } \sum_{j \in N_i^+} x_{ij} - \sum_{j \in N_i^-} x_{ji} = f_i, \\ 0 & \text{if } i \in V(G) \setminus \{s,t\}, \sum_{j \in N_i^+} x_{ij} = \sum_{j \in N_i^-} x_{ji} \text{ and } \sum_{j \in N_i} x_{ij} \le 2, \\ +\infty & \text{otherwise.} \end{cases}$$

Then, solving k-VDSP is equivalent to solving the following factorized optimization problem:

$$\begin{aligned} \min \quad & \sum_{e \in E(G)} \phi_e(x_e) + \sum_{i \in V(G)} \psi_i(x_{E_i}) \\ \text{s.t.} \quad & x_e \in \{0,1\}, \ \forall e \in E(G). \end{aligned}$$

For each arc $e = ij$ on the computation tree, define a message function $m_{e\to j}(x_e)$ as the optimum (min-sum weights) on the subtree below e with e included. Similarly, define $m_{i\to e}(x_e)$ as the optimum on the subtree below i including i but not e. Due to the nature of tree structure, these two message functions can be recursively defined as follows: for any arc $e = ij$,

$$m_{e\to j}(x_e) = \phi_e(x_e) + m_{i\to e}(x_e), \tag{6}$$

$$m_{i\to e}(x_e) = \min_{x_{E_i \setminus e}} \left\{ \psi_i(x_{E_i}) + \sum_{e' \in E_i \setminus e} m_{e'\to i}(x_{e'}) \right\}. \tag{7}$$

Using (1)-(2), starting from leaves, the message functions $m_{e\to j}(x_e)$ and $m_{i\to e}(x_e)$ can be computed for all $e \in E(G), i \in V(G)$. Then, the update messages for each vertex and arc are as follows:

$$m^q_{e\to j}(x_e) = \phi_e(x_e) + m^{q-1}_{i\to e}(x_e),$$

$$m^q_{i\to e}(x_e) = \min_{x_{E_i \setminus e}} \left\{ \psi_i(x_{E_i}) + \sum_{e' \in E_i \setminus e} m^q_{e'\to i}(x_{e'}) \right\}.$$

At the root arc $r = uv$, combine the messages $m_{r\to u}(x_r)$ and $m_{r\to v}(x_r)$, we can derive the estimation at the end of iteration Q on the computation tree T^Q_r as

$$b^Q_r(x_r) = m^Q_{r\to u}(x_r) + m^Q_{r\to v}(x_r) - \phi_r(x_r),$$

which is also known as the *belief* of the root arc. Finally, we describe the Min-Sum BP algorithm for solving k-VDSP in detail as Algorithm 1.

C. Results

Before formally stating our results, we need to define the residual network first with a feasible solution x of k-VDSP. Define G_x to be the residual network of G with respect to a feasible solution x as follow rules:

- G_x has the same vertex set as G, i.e., $V(G) = V(G_x)$;
- For each $e = ij \in E(G)$, if $x_e = 0$, then $e = ij \in E(G_x)$ with weight $w^x_{ij} = w_{ij}$;

Algorithm 1 Min-Sum BP algorithm for k-VDSP

1: Initialize $q = 0$, message $m^0_{i\to e}(x_e) = 0$ for any $e = ij \in E(G)$.
2: **for** $q = 1, 2, ..., Q$ **do**
3: For each $e = ij \in E(G)$, update messages as follows:

$$m^q_{e\to j}(x_e) = \phi_e(x_e) + m^{q-1}_{i\to e}(x_e),$$

$$m^q_{i\to e}(x_e) = \min_{x_{E_i}\setminus e} \left\{ \psi_i(x_{E_i}) + \sum_{e'\in E_i\setminus e} m^q_{e'\to i}(x_{e'}) \right\}.$$

4: $q := q + 1$
5: **end for**
6: For each $e = ij \in E(G)$, set the belief function as

$$b^Q_e(x_e) = m^Q_{e\to i}(x_e) + m^Q_{e\to j}(x_e) - \phi_e(x_e).$$

7: Calculate the belief estimate by finding $\widehat{x}^Q_e \in \arg\min_{x_e\in\{0,1\}} b^Q_e(x_e)$ for each $e \in E(G)$.
8: Return $\widehat{x}^Q = \{\widehat{x}^Q_e : e \in E(G)\}$ as an estimation of the optimal solution.

- For each $e = ij \in E(G)$, if $x_e = 1$, then $e' = ji \in E(G_x)$ with weight $w^x_{ji} = -w_{ij}$.

Let

$$o(x) = \min_{C\in\mathscr{C}} \left\{ w^x(C) = \sum_{e\in C} w^x_e \right\},$$

where $\mathscr{C}$ is the set of directed cycles in G_x. Note that if x^* is the unique optimal solution of k-VDSP in digraph G, then $o(x^*) > 0$ must hold in G_{x^*}, or else we can change x^* along the minimum weight cycle in $o(x^*)$ without increasing its weight.

Theorem 3.1: For any digraph G of order n, if the k-VDSP on G has a unique optimal solution x^*, then Min-Sum BP algorithm converges to x^* within $(\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n$ iterations, where U is the maximum weight of a simple directed path in G_{x^*}. That is to say $\widehat{x}^Q = x^*$ when $Q \geq (\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n$.

Let $w_{max} = \max\{w_e : e \in E(G)\}$. Then by the definition of U, we have that $U \leq nw_{max}$ since simple directed path has at most $n - 1$ arcs in G_{x^*}. If w_e is integral for each $e \in E(G)$, then $o(x^*)$ is also positive integral. It follows that $\lfloor \frac{U}{2o(x^*)} \rfloor \leq nw_{max}$. Combine this with Theorem 3.1, we have the corollary as follows.

Corollary 3.2: For any digraph G of size n, if the problem k-VDSP on G has a unique optimal solution x^*, then Min-Sum BP algorithm converges to x^* within $O(n^2 w_{max})$ iterations, provided that the weight on each arc is nonnegative integral.

IV. Proof of Correctness and Convergence

In this section, we establish the convergence of Min-Sum BP algorithm to the optimal solution of k-VDSP. Before proving Theorem 3.1, we need to show two important lemmas as follows. Loosely speaking, VDSP^Q_r is essentially a k-VDSP on the computation tree T^Q_r: there are similar constraints for any arc $e \in E(T^Q_r)$ and any vertex, except for those at the Q-level.

Lemma 4.1: Let $\widehat{x}^Q_r$ be the value of the output of the BP algorithm at the end of iteration Q on arc $r \in E(G)$. Then there exists an optimal solution y^* of VDSP^Q_r such that $y^*_r = \widehat{x}^Q_r$ where r is the root of computation tree T^Q_r.

Proof: Let $r = uv$ be the root arc of computation tree T^Q_r. By definition, T^Q_r has two components, denoted by C and C', which are connected via the root arc r. Without loss of generality, we assume C is the component containing u. Let $T^Q_{r\to v}$ denote $C \cup r$, which can be viewed as a subtree of T^Q_r. Next, let $V^0(T^Q_{r\to v})$ be the set of all the vertices of $T^Q_{r\to v}$, excluding those at the Q-evel. Recall that $\gamma(s') = s$ and $\gamma(t') = t$, where s, t is the given source and sink, respectively. Denote by $E(T^Q_{r\to v})$ the set of arcs in $T^Q_{r\to v}$. Then we define $\text{VDSP}^Q_{r\to v}(z)$ as follows.

$$
\begin{aligned}
\min \quad & \sum_{e\in E(T^Q_{r\to v})} w_e y_e && && (\text{VDSP}^Q_{r\to v}(z))\\
\text{s.t.} \quad & \sum_{j\in N_i^+(T_r^Q)} y_{ij} - \sum_{j\in N_i^-(T_r^Q)} y_{ji} = f_{\gamma(i)}, && \forall\, i \in V^o(T^Q_{r\to v});\\
& \sum_{j\in N_i^+(T_r^Q)} y_{ij} + \sum_{j\in N_i^-(T_r^Q)} y_{ji} \le 2, && \forall\, i \in V^o(T^Q_{r\to v}) \setminus \{s',t'\};\\
& y_r = z\\
& y_e \in \{0,1\}, && \forall\, e \in E(T^Q_{r\to v}),
\end{aligned}
$$

where $N_i^+(T_r^Q), N_i^-(T_r^Q)$ denote the sets of out-neighbors and in-neighbors of i in T_r^Q, respectively.

Now, we show that under the Min-Sum BP algorithm (running on G) the value of message function $m^Q_{r\to\gamma(v)}(z)$ is the same as the weight of the optimal assignment for $\text{VDSP}^Q_{r\to v}(z)$. This can be established by induction. When $Q = 1$, the statement can be checked to be true trivially. Denote by $E_u(T_r^Q)$ the set of arcs incident to u in T_r^Q. For $Q > 1$ and each $a \in E_u(T_r^Q)\backslash r$ with $a = pu$ (or up), let $T^{Q-1}_{a\to u}$ be the subtree of $T^Q_{r\to v}$ that includes everything in $T^Q_{r\to v}$ but u, v and r. Consider the sub-problem $\text{VDSP}^{Q-1}_{a\to u}(z)$ as follows.

$$
\begin{aligned}
\min \quad & \sum_{e\in E(T^{Q-1}_{a\to u})} w_e y_e && && (\text{VDSP}^{Q-1}_{a\to u}(z))\\
\text{s.t.} \quad & \sum_{j\in N_i^+(T_r^Q)} y_{ij} - \sum_{j\in N_i^-(T_r^Q)} y_{ji} = f_{\gamma(i)}, && \forall\, i \in V^o(T^{Q-1}_{a\to u});\\
& \sum_{j\in N_i^+(T_r^Q)} y_{ij} + \sum_{j\in N_i^-(T_r^Q)} y_{ji} \le 2, && \forall\, i \in V^o(T^{Q-1}_{a\to u}) \setminus \{s',t'\};\\
& y_a = z\\
& y_e \in \{0,1\}, && \forall\, e \in E(T^{Q-1}_{a\to u}).
\end{aligned}
$$

By induction hypothesis, it must be that the value of $m^{Q-1}_{a\to\gamma(u)}(z)$ equals the weight of the solution of $\text{VDSP}^{Q-1}_{a\to u}(z)$. Due to the hypothesis and the relation of subtree $T^{Q-1}_{a\to u}$ for all $a \in E_u(T_r^Q)\backslash r$ with $T^Q_{r\to v}$, it follows that the problem $\text{VDSP}^Q_{r\to v}(z)$ is equivalent to

$$
\begin{aligned}
\min \quad & w_r z + \sum_{a\in E_u(T_r^Q)\backslash r} m^{Q-1}_{a\to\gamma(u)}(x_a)\\
\text{s.t.} \quad & \sum_{v\in N_u^+(T_r^Q)} y_{uv} - \sum_{v\in N_u^-(T_r^Q)} y_{vu} = f_{\gamma(u)},\\
& \sum_{v\in N_u^+(T_r^Q)} y_{uv} + \sum_{v\in N_u^-(T_r^Q)} y_{vu} \le 2, && if\ u \notin \{s',t'\};\\
& y_r = z,\\
& y_a \in \{0,1\}, && \forall\, a \in E_u(T_r^Q)\backslash r.
\end{aligned}
$$

This is exactly the same as the relation between $m^Q_{r\to\gamma(v)}(z)$ and message function $m^{Q-1}_{a\to\gamma(u)}(\cdot)$ for $a \in E_u(T_r^Q)\backslash r$ as

$$
m^Q_{r\to\gamma(v)}(z) = w_r z + \sum_{a\in E_u(T_r^Q)\backslash r} m^{Q-1}_{a\to\gamma(u)}(x_a).
$$

That is, $m^Q_{r\to\gamma(v)}(z)$ is exactly the same as the weight of optimal assignment of $\text{VDSP}^Q_{r\to v}(z)$. Using this equivalence, we will complete the proof of Lemma 4.1.

Finally, for given $r = uv$, the problem $\text{VDSP}^Q_r(z)$ is equivalent to

$$\begin{aligned}
\min \quad & -w_r z + \sum_{e\in E(T^Q_{r\to u})} w_e y_e + \sum_{e\in E(T^Q_{r\to v})} w_e y_e \\
\text{s.t.} \quad & \sum_{j\in N_i^+(T^Q_r)} y_{ij} - \sum_{j\in N_i^-(T^Q_r)} y_{ji} = f_{\gamma(i)}, && \forall\ i \in V^o(T^Q_r); \\
& \sum_{j\in N_i^+(T^Q_r)} y_{ij} + \sum_{j\in N_i^-(T^Q_r)} y_{ji} \le 2, && \forall\ i \in V^o(T^Q_r)\setminus\{s',t'\}; \\
& y_e \in \{0,1\}, && \forall\ e \in E(T^Q_{r\to i}) \cup E(T^Q_{r\to j}).
\end{aligned}$$

That means the min-sum weights of an optimal solution of the problem $\text{VDSP}^Q_r(z)$ equals $m^Q_{r\to\gamma(u)}(z) + m^Q_{r\to\gamma(v)}(z) - w_r z$ for any $z \in \{0,1\}$. Now the claim of Lemma 4.1 follows immediately. ■

Lemma 4.1 exhibits the relation between BP algorithm and computation tree. Next, we prove our main technical lemma which is a key to the proof of Theorem 3.1.

Lemma 4.2: Let x^* be the unique optimal solution of k-VDSP on G. If y^* is the optimal solution of VDSP^Q_r and $Q \ge (\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n$, then we have $y^*_r = x^*_r$ where r is the root of T^Q_r.

Proof: Suppose on the contrary that there is an arc $r_0 = uv \in E(G)$ such that $y^*_{r_0} \ne x^*_{r_0}$. Let $\Lambda^* = \{e \in E(G) : x^*_e = 1\}$ and $\Omega^* = \{e \in E(T^Q_{r_0}) : y^*_e = 1\}$. Without loss of generality, we assume $y^*_{r_0} > x^*_{e_0}$, i.e., $x^*_{r_0} = 0$ and $y^*_{r_0} = 1$. Then, by the definition of Λ^* and Ω^*, we have that $r_0 \in \Omega^* - \Lambda^*$. If a feasible solution of $\text{VDSP}^Q_{r_0}$ can be obtained by modifying y^* such that its total weight strictly less than that of y^*, then a contradiction to the optimality of y^* arises and Lemma 4.2 is established.

Let $r_0 = uv$ be the root of the computation tree $T^Q_{r_0}$ as above. We will choose an arc $r_1 \ne r_0$ incident to u in $T^Q_{r_0}$ as the following rules:

- If $\sum_{j\in N_u(T^Q_{r_0})} x_{ju} = 0$, then there exists an in-arc r_1 for u such that $y^*_{r_1} = 1$;
- Otherwise, there exists an out-arc r_1 for u such that $x^*_{r_1} = 1$.

Similarly, we can choose an arc $r_{-1} \ne r_0$ incident to v in $T^Q_{r_0}$ as the following rules:

- If $\sum_{j\in N_v(T^Q_{r_0})} x_{jv} = 0$, then there exists an out-arc r_{-1} for v such that $y^*_{r_{-1}} = 1$;
- Otherwise, there exists an in-arc r_{-1} for v such that $x^*_{r_{-1}} = 1$.

Let u_1, v_1 be the other ends of r_1, r_{-1}, respectively. Then we can apply recursively the similar reasoning for u_1 and v_1 so that the feasibility condition of x^*, y^* and the inequalities between the value of components of x^*, y^* at arcs r_1, r_{-1} lead to the existence of arcs r_2, r_{-2} incident to u_1, v_1, respectively. Continuing this manner all the way down to the leaves, we will find a path starting and ending in leaves of $T^Q_{r_0}$, denoted by $P = \{r_{-Q}, ..., r_{-1}, r_0, r_1, ..., r_Q\}$, such that for $-Q \le l \le Q$,

$$\begin{aligned}
r_l \in \Omega^* - \Lambda^* \quad &\Leftrightarrow \quad \text{both } r_l \text{ and } r_0 \text{ have the same orientation}, \\
r_l \in \Lambda^* - \Omega^* \quad &\Leftrightarrow \quad \text{both } r_l \text{ and } r_0 \text{ have the opposite orientation}.
\end{aligned}$$

Figure 2 demonstrates this path P with dashed arcs.

Now, we can modify y^* to obtain a new feasible solution $\widetilde{y}$ of VDSP^Q_r as following. Let $\Omega' = (\Omega^* - \Omega^* \cap P) \cup (\Lambda^* \cap P)$ and $\widetilde{y}$ be the solution of VDSP^Q_r corresponding to Ω'. Furthermore, for any vertex i on P, let r' and r'' be the arcs which are incident to i and belong to P. Then we have that for any vertex i on P,

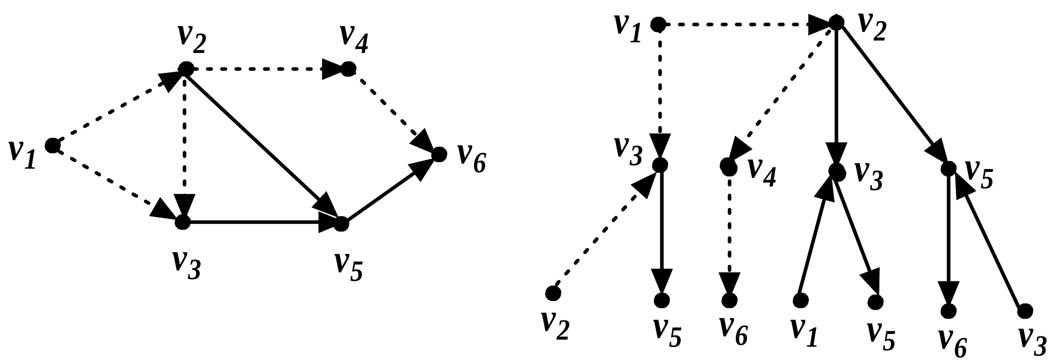

Fig. 2: An example of the path P on a computation tree $T^2_{v_1v_2}$ with dashed arcs.

- if r' and r'' have the same orientation as r_0, then

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} - \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = (-1 + \sum_{j\in N_i^+(T_r^Q)} y_{ij}) - (-1 + \sum_{j\in N_i^-(T_r^Q)} y_{ji}) = f_{\gamma(i)},$$

  and

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} + \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = (-1 + \sum_{j\in N_i^+(T_r^Q)} y_{ij}) + (-1 + \sum_{j\in N_i^-(T_r^Q)} y_{ji}) = (-1+1)+(-1+1) = 0 \le 2.$$

- if r' and r'' have the opposite orientation as r_0, then

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} - \sum_{j\in N_i^-} \widetilde{y}_{ji} = (1 + \sum_{j\in N_i^+(T_r^Q)} y_{ij}) - (1 + \sum_{j\in N_i^-(T_r^Q)} y_{ji}) = f_{\gamma(i)},$$

  and

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} + \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = (1 + \sum_{j\in N_i^+(T_r^Q)} y_{ij}) + (1 + \sum_{j\in N_i^-(T_r^Q)} y_{ji}) = (1+0)+(1+0) = 2 \le 2.$$

- if r' has the same orientation and r'' has the opposite orientation as r_0, then both r' and r'' are in-arcs or out-arcs for i.

  (1) If both r' and r'' are in-arcs for i, then

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} - \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = \sum_{j\in N_i^+(T_r^Q)} y_{ij} - (-1 + 1 + \sum_{j\in N_i^-(T_r^Q)} y_{ji}) = f_{\gamma(i)},$$

  and

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} + \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = \sum_{j\in N_i^+(T_r^Q)} y_{ij} + (-1 + 1 + \sum_{j\in N_i^-(T_r^Q)} y_{ji}) \le 2;$$

  (2) If both r' and r'' are out-arcs for i, then

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} - \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = (-1 + 1 + \sum_{j\in N_i^+(T_r^Q)} y_{ij}) - \sum_{j\in N_i^-(T_r^Q)} y_{ji} = f_{\gamma(i)},$$

  and

$$\sum_{j\in N_i^+(T_r^Q)} \widetilde{y}_{ij} + \sum_{j\in N_i^-(T_r^Q)} \widetilde{y}_{ji} = (-1 + 1 + \sum_{j\in N_i^+(T_r^Q)} y_{ij}) + \sum_{j\in N_i^-(T_r^Q)} y_{ji} \le 2.$$

This implies $\widetilde{y}$ satisfies all the other equality constraints of VDSP$_r^Q$, since only the values of the arcs in P are changed. Therefore, $\widetilde{y}$ is a feasible solution of VDSP$_r^Q$.

Recall that $P = \{r_{-Q}, ..., r_{-1}, r_0, r_1, ..., r_Q\}$. For any $r_l = ij \in P$, we define $\widetilde{r_l} = ij$ if $x^*_{r_l} = 0$, and $\widetilde{r_l} = ji$ if $x^*_{r_l} = 1$, where $-Q \leq l \leq Q$. Let $\widetilde{P} = \{\widetilde{r}_{-Q}, ..., \widetilde{r}_{-1}, \widetilde{r}_0, \widetilde{r}_1, ..., \widetilde{r}_Q\}$. Given the value of x^* and definition of $\widetilde{r_l}$, it can be checked that $\widetilde{r_l}$ is an arc in the residual network G_{x^*}, and $\widetilde{P}$ is a directed walk in G_{x^*}. Then $\widetilde{P}$ can be decomposed into a simple directed path D and a collection of simple directed cycles $C_1, ..., C_d$. Note that each simple directed cycle or path on G_{x^*} can have at most n arcs. Since there are $2Q+1$ arcs in $\widetilde{P}$ and $Q \geq (\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n$, we have

$$d > \frac{2Q+1}{n} \geq \frac{2(\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n + 1}{n} > \frac{U}{o(x^*)}.$$

Then we can obtain that the weight of $\widetilde{P}$ is strictly positive:

$$w(\widetilde{P}) = w(D) + \sum_{i=1}^{m} w(C_i) \geq -U + d \cdot o(x^*) > -U + \frac{U}{o(x^*)} o(x^*) = 0,$$

where $w(\widetilde{P}), w(D), w(C_i)$ denote the sum of weights of all the arcs in $w(\widetilde{P}), w(D), w(C_i)$, respectively.

Finally, for any $Q \geq (\lfloor \frac{U}{2o(x^*)} \rfloor + 1)n$, we have

$$\begin{aligned}
\sum_{e \in E(T_r^Q)} w_e y_e^* - \sum_{e \in E(T_r^Q)} w_e \widetilde{y}_e &= \sum_{e \in E(T_r^Q)} w_e (y_e^* - \widetilde{y}_e) \\
&= \sum_{e \in \Omega^* \cap P} w_e - \sum_{e \in \Lambda^* \cap P} w_e \\
&= \sum_{e \in \widetilde{P}} w_e \\
&= w(\widetilde{P}) \\
&> 0.
\end{aligned}$$

The last inequality leads a contradiction that y^* is the optimal solution of VDSP$_r^Q$ which completes the proof. ■

Now we can complete the proof of Theorem 3.1 and establish the correctness and convergence of Min-Sum BP for k-VDSP as follows.

Proof of Theorem 3.1: Suppose to the contrary that there exists $r \in E(G)$ and $Q \geq (\lfloor \frac{U}{2o(X^*)} \rfloor + 1)n$ such that $\widehat{x}_r^Q \neq x_r^*$. According to the relation between BP and computation tree T_r^Q as Lemma 4.1, there is an optimal solution y^* of VDSP$_r^Q$ such that $y_r^* = \widehat{x}_r^Q$ when r is the root of T_r^Q. Then we have $y_r^* \neq x_r^*$ which contradicts Lemma 4.2. Therefore, the assumption that $\widehat{x}_r^Q \neq x_r^*$ does not hold. This completes the proof of Theorem 3.1.■

V. Extensions

We now establish the extensions of k-VDSP to the versions of multiple sources or sinks. The main ideas remain unchanged, and thus the proofs are omitted here. The key differences in mathematical programming between the problem k-VDSP and its versions are the definition of vertex demand function $f : V(G) \to \mathbb{Z}$ for some $i \in V(G)$.

A. The version of multiple sources

For k given sources $S := \{s_1, s_2, ..., s_k\} \subseteq V(G)$ and a sink $t \in V(G)$, it aims to compute k vertex-disjoint (besides t) paths $P_1, P_2, ..., P_k$ in G, such that $\sum_{i=1}^{k} w(P_i)$ attains the minimum. Define the function $f : V(G) \to \mathbb{Z}$ that $f_t = -k$, $f_i = 1$ for any $i \in S$ and $f_i = 0$ for any $i \in V(G) \setminus (\{t\} \cup S)$. The version of multiple sources is given by the following integer program:

$$\begin{aligned}
\min \quad & \sum_{e \in E(G)} w_e x_e \\
\text{s.t.} \quad & \sum_{j \in N_i^+} x_{ij} - \sum_{j \in N_i^-} x_{ji} = f_i, \ \forall\, i \in V(G); \\
& \sum_{j \in N_i^+} x_{ij} + \sum_{j \in N_i^-} x_{ji} \leq 2, \ \forall\, i \in V(G) \setminus (\{t\} \cup S); \\
& x_e \in \{0, 1\}, \quad \forall\, e \in E(G).
\end{aligned}$$

B. The version of multiple sinks

For a given a source $s \in V(G)$ and k sinks $T := \{t_1, t_2, ..., t_k\} \subseteq V(G)$, it aims to compute k vertex-disjoint (besides s) paths $P_1, P_2, ..., P_k$ in G, such that $\sum_{i=1,2,...,k} w(P_i)$ attains the minimum. Define the function $f : V(G) \to \mathbb{Z}$ that $f_s = k$, $f_i = -1$ for any $i \in T$ and $f_i = 0$ for any $i \in V(G) \setminus (\{s\} \cup T)$. The version of multiple sinks is given by the following integer program:

$$\begin{aligned}
\min \quad & \sum_{e \in E(G)} w_e x_e \\
\text{s.t.} \quad & \sum_{j \in N_i^+} x_{ij} - \sum_{j \in N_i^-} x_{ji} = f_i, \ \forall\, i \in V(G); \\
& \sum_{j \in N_i^+} x_{ij} + \sum_{j \in N_i^-} x_{ji} \leq 2, \ \forall\, i \in V(G) \setminus (\{s\} \cup T); \\
& x_e \in \{0, 1\}, \quad \forall\, e \in E(G).
\end{aligned}$$

C. The version of multiple sources and multiple sinks

For k given sources $\{s_1, s_2, ..., s_k\} \subseteq V(G)$ and k sinks $\{t_1, t_2, ..., t_k\} \subseteq V(G)$, it aims to compute k vertex-disjoint paths $P_1, P_2, ..., P_k$ in G, such that $\sum_{i=1}^{k} w(P_i)$ attains the minimum. Let $S := \{s_1, s_2, ..., s_k\}$, $T := \{t_1, t_2, ..., t_k\}$. Define the function $f : V(G) \to \mathbb{Z}$ that $f_i = 1$ for any $i \in S$, $f_i = -1$ for any $i \in T$ and $f_i = 0$ for any $i \in V(G) \setminus (S \cup T)$. The version of k sources and k sinks is given by the following integer program:

$$\begin{aligned}
\min \quad & \sum_{e \in E(G)} w_e x_e \\
\text{s.t.} \quad & \sum_{j \in N_i^+} x_{ij} - \sum_{j \in N_i^-} x_{ji} = f_i, \ \forall\, i \in V(G); \\
& \sum_{j \in N_i^+} x_{ij} + \sum_{j \in N_i^-} x_{ji} \leq 2, \ \forall\, i \in V(G) \setminus (S \cup T); \\
& x_e \in \{0, 1\}, \quad \forall\, e \in E(G).
\end{aligned}$$

VI. Conclusions

In this paper, we formulated the Min-Sum BP algorithm for the vertex-disjoint shortest k-path problem (k-VDSP) and analyzed the correctness and convergence of the algorithm presented. We established that the Min-Sum BP algorithm solves k-VDSP exactly in $O(n^2 w_{max})$ iterations, provided that the optimal solution is unique and the weight parameter is nonnegative integral. Although the running time of our algorithm for k-VDSP is not better than that of other existing algorithms for k-VDSP, the advantage of message-passing algorithms based on BP is that it is widely applicable and easy to implement for a broad class of constrained optimization problems. Due to its distributed nature, the BP algorithm and its variants can also run fast on a large data network in synchronous circumstances.

Acknowledgements

We are very grateful to the reviewers for their invaluable suggestions and comments, which greatly help to improve the manuscript. The research was partially supported by National Natural Science Foundation of China (Grant Nos. 11871280, 11971349, 61772005 and U1811461), the Natural Science Foundation of Guangdong Province (Grant No. 2020B1515310009) and Qinglan Project of Jiangsu Province.

References

[1] D. Achlioptas and F. Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In Proceedings of the thirty-eighth annual ACM symposium on theory of computing, 2006, pages 130-139.

[2] J. Bang-Jensen, G. Gutin and A. Yeo. When the greedy algorithm fails. Discrete Optimization, 1:121-127, 2004.

[3] M. Bayati, C. Borgs, J. Chayes, and R. Zecchina. Belief propagation for weighted b-matchings on arbitrary graphs and its relation to linear programs with integer solutions. SIAM Journal on Discrete Mathematics, 25:989-1011, 2011.

[4] M. Bayati, D. Shah, and M. Sharma. Max-product for maximum weight matching: convergence, correctness, and LP duality. IEEE Transactions on Information Theory, 54:1241-1251, 2008.

[5] Y. Cheng, M. Neely, and K. M. Chugg. Iterative message passing algorithm for bipartite maximum weighted matching. In Proceedings of IEEE International Symposium Information Theory, Cambridge, 2006, pages 1934-1938.

[6] G. Dai, F. Li, Y. Sun, D. Xu, and X. Zhang. Convergence and correctness of belief propagation for the Chinese postman problem. Journal of Global Optimization, 75:813-831, 2019.

[7] G. Even and N. Halabi. Analysis of the Min-Sum Algorithm for Packing and Covering Problems via Linear Programming. IEEE Transactions on Information Theory, 61:5295-5305, 2015.

[8] U. Feige, E. Mossel and D. Vilenchik. Complete convergence of message passing algorithms for some satisfiability problems. Approximation, Randomization, and Combinatorial Optimization, Algorithms and Techniques, 2006, pages 339-350.

[9] B. J. Frey and D. Dueck. Clustering by passing messages between data points. Science, 315:972-976, 2007.

[10] G. Gutin and A. Punnen. The traveling salesman problem and its variations. Kluwer, Dordrecht, 2002.

[11] G. Gutin, A. Yeo and A. Zverovich. Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP. Discrete Applied Mathematics, 117:81-86, 2002.

[12] R. G. Gallager. Low density parity check codes. Cambridge, 1963, pages 21-28.

[13] D. Gamarnik, D. Shah, and Y. Wei. Belief propagation for min-cost network flow: convergence and correctness. Operations Research, 60:410-428, 2012.

[14] G.B. Horn. Iterative Decoding and Pseudocodewords. Ph.D. dissertation, Dep.Elec.Eng., California Inst. Technol., Pasadena, CA, 1999.

[15] A. Itai, Y. Peal and Y. Shiloach, The complexity of finding maximum disjoint paths with length constraints, Networks, 12:277-286, 1982.

[16] F.R. Kschischang, B.J. Frey and H.A. Loeliger. Factor graphs and sum-product algorithm. IEEE Transactions on Information Theory, 47:498-519, 2001.

[17] M. Mézard. Passing messages between disciplines. Science, 301:1685-1686, 2003.

[18] M. Mézard, G. Parisi and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. Science, 297:812-815, 2002.

[19] M. Mézard and R. Zecchina. Random k-satisfiability problem: from an analytic solution to an efficient algorithm. Physical Review, 66:249-264, 2002.

[20] J. Pearl. Probabilistic reasoning in intelligent systems: networks of plausible reasoning. Morgan Kaufmann, 1988.

[21] T. Richardson and R. Urbanke. The capacity of low-density parity check codes under message-passing decoding. IEEE Transactions on Information Theory, 47:599-618, 2001.

[22] S. Sanghavi, D. Malioutov, and A. Willsky. Belief propagation and LP relaxation for weighted matching in general graphs. IEEE Transactions on Information Theory, 54:2203-2212, 2011.

[23] S. Sanghavi, D. Shah, A. S. Willsky. Message passing for maximum weight independent set. IEEE Transactions on Information Theory, 55:4822-4834, 2009.

[24] J.W. Suurballe. Disjoint paths in a network. Networks, 4(2):125-145, 1974.

[25] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. Exploring Artificial Intelligence in the New Millennium, 8:236-239, 2003.