JOURNAL OF INDUSTRIAL AND
MANAGEMENT OPTIMIZATION

doi:10.3934/jimo.2021116

APPROXIMATION ALGORITHM WITH CONSTANT RATIO FOR STOCHASTIC PRIZE-COLLECTING STEINER TREE PROBLEM

Jian Sun

Department of Operations Research and Information Engineering
Beijing University of Technology
Beijing 100124, China

Haiyun Sheng

School of Mathematical Science & Institute of Mathematics
Nanjing Normal University
Jiangsu 210023, China

Yuefang Sun

School of Mathematics and Statistics
Ningbo University
Zhejiang 315211, China

Donglei Du

Faculty of Management
University of New Brunswick
New Brunswick, E3B 5A3, Canada

Xiaoyan Zhang*

School of Mathematical Science & Institute of Mathematics
Nanjing Normal University
Jiangsu 210023, China

(Communicated by Wenxun Xing)

Abstract. Steiner tree problem is a typical NP-hard problem, which has vast application background and has been an active research topic in recent years. Stochastic optimization problem is an important branch in the field of optimization. Compared with deterministic optimization problem, it is an optimization problem with random factors, and requires the use of tools such as probability and statistics, stochastic process and stochastic analysis. In this paper, we study a two-stage finite-scenario stochastic prize-collecting Steiner tree problem, where the goal is to minimize the sum of the first stage cost, the expected second stage cost and the expected penalty cost. Our main contribution is to present a primal-dual 3-approximation algorithm for this problem.

2020 *Mathematics Subject Classification.* Primary: 90C27, 68W25.

Key words and phrases. Prize-collecting Steiner tree, combinatorial optimization, stochastic optimization, approximation algorithm.

A preliminary version appeared in the Proceedings of the 13th International Conference on Algorithmic Aspects in Information and Management.

* Corresponding author: Xiaoyan Zhang, zhangxiaoyannjnu@126.com.

1. **Introduction.** The Steiner tree problem (STP) is a fundamental and extensively investigated problem in combinatorial optimization. In this problem, we are given an undirected graph $G = (V, E)$ with an edge-cost function $c : E \to \mathbb{R}_{\geq 0}$, and a set of terminals $\emptyset \neq T \subseteq V$. The aim is to find a tree which spans a given terminal set T with minimum cost. Each non-terminal vertex contained in the Steiner tree is called a Steiner point. The STP is NP-hard, even in the Euclidean or rectilinear metrics [15]. Therefore, one approach for solving the STP (and other NP-hard problems) is to design a (polynomial-time) approximation algorithm whose performance is measured by the approximation ratio. Another common approach is heuristic: any algorithm based on intuitive or empirical construction that presents a feasible solution to each instance of a combinatorial optimization problem to be solved at an acceptable cost (i.e., computational time and space), but the degree of deviation from the optimal solution cannot generally be predicted.

For decades, the most obvious algorithm for the Steiner tree problem has been the minimum spanning tree heuristic (MST heuristic), which approximates a Steiner minimum tree of T with the minimum spanning tree [16]. In other words, the original Steiner tree problem is reduced to the minimum spanning tree problem in the graph induced by the set of terminal nodes T. Zelikovsky designed a 11/6-approximation algorithm for Steiner tree problem in graphs [30]. Based on a novel iterative randomized rounding technique, Byrka et. al. proposed the best known approximation algorithm for the STP with ratio $\ln 4 + \varepsilon \approx 1.39$ [8]. On a high level, all these results in [24, 25, 27, 30] build upon the notion of the k-restricted Steiner tree. For a tree, if every leaf is a terminal vertex, then it is called a component. The k-restricted Steiner tree S is a set of components, each of which contains at most k terminals (k-components), whose union constitutes the Steiner tree (see [6] for detailed introduction).

Gilbert and Pollak [16]	2	SIAM J. Appl. Math. 1968
Zelikovsky [30]	$\frac{11}{6}$	Algorithmica 1993
Karpinski and Zelikovsky [24]	1.644	JOCO 1997
Prömel and Steger [25]	$\frac{5}{3}$	J. Algorithm 2000
Robin and Zelikovsky [27]	1.55	SIAM J. Disc. Math. 2005
Byrka et. al. [8]	$\ln 4 + \epsilon$	J. ACM 2013

TABLE 1. Approximation guarantees for STP

For the the case of edge lengths 1 and 2, Berman et al. presented a 1.25-approximation algorithm [3]. In the special case that G is a planar graph, Borradaile et al. obtained a polynomial time approximation scheme for the Steiner tree problem [7].

The prize-collecting Steiner tree problem is an extension of the Steiner tree problem, where we can refuse to service certain vertices, but pay a penalty if the service is too expensive. The aim is to find a tree that minimizes the sum of the cost of its edges and the penalties of vertices not included in the tree. This problem has many applications in network design and has been used to approximate many other problems. The first approximation algorithm for the prize-collecting Steiner tree problem (PCST) was given by Bienstock et al. [5]. The natural linear programming (LP) relaxation of PCST has an integrality gap of 2, which has been a barrier to further improvements for this problem. Archer et al. presented $(2 - \epsilon)$-approximation algorithms for this problems, connected by a unified technique for

improving prize-collecting algorithms that circumvented the integrality gap barrier [1].

The primal-dual scheme is a useful technique in designing approximation algorithms for NP-hard problems [9, 13, 21, 29]. Goemans and Williamson [17] used the primal-dual scheme to derive a 2-approximation for the rooted PCST, where $n = |V|$. For the unrooted PCST, they also presented a 2-approximation by trying all possible choices for the root. For the sake of convenience, this algorithm is denoted as GW-algorithm. Johnson, Minkoff and Phillips improved the GW-algorithm in terms of time complexity by proposing a variant of the GW-algorithm that runs the primal-dual scheme only once, resulting in a $O(n^2 logn)$ time bound [22]. Cole et al. proposed a faster implementation of the GW-algorithm, which also runs the primal-dual scheme only once and produces a $(2 + \frac{1}{ploy(n)})$-approximation for the PCST [12].

So far, the optimization problems under uncertainty have been solved by several complementary modelling paradigms, which are mainly different from the expressions of uncertainty. For instance, stochastic programming makes an assumption on the uncertainty that it is controlled by a known probability distribution and attempts to minimize certain quantity of the probability functions such as expected costs or quantiles of cost distributions. In contrast with the stochastic programming, robust optimization ignores all distribution information, and aims to minimize the worst-case cost under all possible uncertainties. Stochastic programming may rely on distribution information that is unavailable or difficult to obtain in practice, while robust optimization models may be too pessimistic about uncertainty, leading to over-conservative decisions.

Stochastic combination optimization problems are usually identified as preplanning problems, i.e., purchasing and allocating resources to satisfy unclear needs when solving problems. For example, network designers may need to make best guess about the future needs of the network and purchase capabilities accordingly. However, it is possible to "wait and see" the changes sometimes, or it is possible to postpone decisions about resource allocation until the requirements or constraints become clear. Specifically, in the field of stochastic combination optimization, the input of the problem we consider is uncertain in itself, but it is drawn from a probability distribution, which is provided as an input, and our goal is to search for strategies that minimizes the expected cost.

The two-stage model with recourse is one of the most commonly used models in stochastic optimization. At the beginning, some data information may be deterministic, and the uncertain future is only characterized by probability distribution. The decision made at this time is called the first-stage decision. Subsequently, there may be an opportunity to enhance the solution of the first stage so as to optimize the realized scenario when realizing the actual future. The second stage of decision-making is called the recourse stage. The goal is to optimize the decision variables of the first stage, so that we can minimize the expected cost of the two-stages.

Formally, we define a two-stage stochastic optimization problem with recourse as follows. In the first stage, there is only partial information available, the policy-maker has to select a set of decision variables, denoted by x^0, based on these information. Later, in order to augment the first stage solution x^0, some second stage variables x^1 are selected with full information provided.

In order to write the stochastic program, we define some variables and symbols as follows: Define ζ to be the random vector, furthermore when the full information is

available, the constraint matrix, the cost vector and the requirement vector defined by ζ are denoted by T, q and h, respectively. In the first stage, we define A, c, and b for the same purpose. P represents the additional constraints that x^0 and x^1 need to satisfy, such as nonnegativity or integrality. Then the stochastic program can be written according to [26] as follows:

$$\begin{aligned} \min \quad & c^\top x^0 + E_\zeta Q(x^0, \zeta) \qquad (SP1) \\ \text{s.t.} \quad & Ax^0 = b, \\ & x^0 \in P, \end{aligned}$$

where

$$\begin{aligned} Q(x^0, \zeta) = \min \, & q^\top x^1 \\ \text{s.t.} \quad & T(x^0, x^1) = h, \\ & x^1 \in P. \end{aligned}$$

Here $Q(x^0, \zeta)$ denotes the optimal cost of the second stage, which is conditioned on scenario $\zeta = (q, T, h)$ having been realized and a first stage setting of the variables x^0. The expectation is taken in terms of ζ.

Subsequently, we consider the two-stage stochastic optimization problem with recourse and an additional restriction. Specifically, the two-stage model is characterized by a finite set of m scenarios in the second stage. The probability of the scenario k occurring is p_k, besides let T^k denote the constraint matrix, q^k be the cost vector and h^k denote the requirement vector in the scenario k respectively. In this restricted case, $SP1$ can be rewritten as follows:

$$\begin{aligned} \min \quad & c^\top x^0 + \sum_{k=1}^{m} p_k (q^k)^\top x^k \qquad (SP2) \\ \text{s.t.} \quad & Ax_0 = b, \\ & T^k(x^0, x^k) = h^k, && \forall k = 1, 2, ..., m, \\ & (x^0, x^k) \in P, && \forall k = 1, 2, ..., m, \end{aligned}$$

in which x^k represents our choice if scenario k materializes.

The two-stage stochastic Steiner tree problem is a natural extension of the STP to a two-stage stochastic combinatorial optimization problem. In the first stage, although both terminal set and edge costs are uncertain, it is still possible to purchase some profitable edges. In fact, all of the possible outcomes are known, and consist of a set of scenarios. In the second stage, one scenario is realized, and additional edges are installed in order to connect a known set of terminals in the future. The goal is to determine the edges that need to be purchased in the first stage and in each scenario, so that the terminal set in each scenario is connected and the expected cost of the entire solution is minimized. Gupta and Pál presented a 4-approximation algorithm via boosted sampling technique [18] for this problem.

We focus on a special case of the stochastic prize-collecting Steiner tree problem where clients (terminal vertices) in each scenario may not be served by the purchased edge, but unserved clients (terminal vertices) will be penalized. In order to solve this problem, we formulate it as a 0-1 integer program. Secondly, we also give linear integer relaxation and the corresponding dual linear program. At last, we design a primal-dual algorithm and prove that its approximate ratio is 3.

The remainder of our paper is organized as follows. In Section 2, we introduce the stochastic prize-collecting Steiner tree problem. Furthermore, we propose an algorithm for the problem based on the primal-dual method. In Section 3, we present our main result and prove that the approximation ratio of this algorithm is 3.

2. Stochastic prize-collecting Steiner tree problem.

2.1. Introduction of the problem.

In this section, we will introduce the stochastic prize-collecting Steiner tree problem (SPCSTP) and present the specific relevant mathematical program. There are two-stages in the SPCSTP:

- In the first stage, there is a potential edge set E, it is allowed to choose some edges for serving any client (terminal vertex) that may appear later.
- In the second stage, all possible scenarios and the associated probabilities become known. Note that the number of scenarios may be on an exponential scale.

While we only consider the special case of the SPCSTP in this paper: the number m of the scenarios is polynomial with respect to the input of the problem, that is to say there are polynomial scenarios in the second stage. For a scenario $k \in \{0, 1, 2, ..., m\}$ in the second stage, we define some notations described in Table 2.

p_k	probability of scenario k being realized
c_e^k	purchasing cost of edge e in scenario k
D_k	client (terminal vertex) set in scenario k
$h_k(T_k)$	penalty cost for the unserved client set $T_k \subseteq D_k \setminus r$
(e, k)	an edge-scenario pair ($e \in E$)
(S, k)	active vertex subset-scenario pair in the k-th scenario ($S \subset V$)
$\mathcal{E} := \{(e, k) : e \in E, k = 0, 1, \cdots, m\}$	edge-scenario pair set
$\mathcal{C} := \{(S, k) : S \subset V, k = 0, 1, \cdots, m\}$	vertex subset-scenario pair set
$\mathcal{C}_k := \{(S, k) : S \subset V\}$	the set of the vertex subset-scenario pair (S, k) in scenario k
$\widehat{E}_0$	the temporarily purchased edge in the first stage
$\widehat{E}_k$	the temporarily purchased edge in the second stage with respect to k-th scenario
$\widehat{T}_k$	the penalty client (terminal vertex) set in the k-th scenario

TABLE 2. Explanation to the notations

In this problem, each client (terminal vertex) is either served by an edge purchased in the first stage or the corresponding scenario realized in the second stage; otherwise the client will be unserved and receive the corresponding compensation (penalty cost paid by the decision maker).

The task is to determine the set of edge-vertex subset pairs $\widehat{E}_0$ and $\widehat{E}_k$ to be purchased in the first stage and in the k-th scenario of the second stage ($k = 1, 2, ..., m$) respectively, the set of client-scenario pairs $\widehat{T}_k$ ($k = 1, 2, ..., m$) that will

incur penalties, so as to minimize the sum of the expected edge purchasing cost $\sum_{e\in\widehat{E}_0} c_e^0 + \sum_{k=1}^m \sum_{e\in\widehat{E}_k} c_e^k$ and expected penalty cost $\sum_{k=1}^m p_k h_k(\widehat{T}_k)$.

For ease of expression, set $p_0 = 1$ and $\sum_{k=1}^m p_k = 1$, then we can formulate the SPCSTP as follows:

$$
\begin{aligned}
\min \quad & \sum_{(e,k)\in E} p_k c_e^k x_e^k + \sum_{k=1}^m \sum_{T_k\subseteq D_k\setminus r} p_k h_k(T_k) z_{T_k} \\
(IP) \quad \text{s.t.} \quad & \sum_{e\in\delta(S)} x_e^0 + \sum_{e\in\delta(S)} x_e^k + \sum_{T_k: S\subseteq T_k} z_{T_k} \geq 1, && \forall (S,k)\in\mathcal{C}, r\notin S, \\
& \sum_{T_k: T_k\subseteq D_k\setminus r} z_{T_k} \leq 1, \\
& x_e^0, x_e^k, z_{T_k} \in \{0,1\}, && \forall e\in E, T_k\subseteq D_k\setminus r.
\end{aligned}
$$

For the sake of understanding the above programming, we will make some explanation on the variables and constraints. For a scenario $k \in \{1, 2, \ldots, m\}$, the penalty cost h_k is a linear function, i.e., $h_k(T_k) = \sum_{i\in T_k} h_k(i)$. $x_e^0 = 1$ indicates the edge $e \in E$ is purchased in the first stage (i.e. edge-scenario pair $(e, 0)$ is purchased); otherwise, $x_e^0 = 0$. Similarly, x_e^k indicates whether edge e is purchased in the k-th scenario of the second stage, if $x_e^k = 1$, edge $e \in E$ is purchased in the second stage for the k-th scenario (i.e. edge-scenario pair (e, k) is purchased); otherwise, $x_e^k = 0$. z_{T_k} represents whether a set of clients $T_k \subseteq D_k \setminus r$ incurs penalties or not. The first constraint models that either there is an edge in $\delta(S)$, or S is a part of a vertex set that pays penalty (a vertex set T_k with $z_{T_k} = 1$) for each client-scenario pair (S, k).

The following LP relaxation and the corresponding dual linear program can be obtained by relaxing the integrality constraints:

$$
\begin{aligned}
\min \quad & \sum_{(e,k)\in E} p_k c_e^k x_e^k + \sum_{k=1}^m \sum_{T_k\subseteq D_k\setminus r} p_k h_k(T_k) z_{T_k} \\
(LP) \quad \text{s.t.} \quad & \sum_{e\in\delta(S)} x_e^0 + \sum_{e\in\delta(S)} x_e^k + \sum_{T_k: S\subseteq T_k} z_{T_k} \geq 1, && \forall (S,k)\in\mathcal{C}, r\notin S, \\
& x_e^0, x_e^k, z_{T_k} \geq 0, && \forall e\in E, T_k\subseteq D_k\setminus r.
\end{aligned}
$$

$$
\begin{aligned}
\max \quad & \sum_{(S,k)\in\mathcal{C}: r\notin S} y_S^k \\
(DP) \quad \text{s.t.} \quad & \sum_{(S,k)\in\mathcal{C}: r\notin S} y_S^k \leq c_e^0, && \forall e\in E, \\
& \sum_{S\subseteq D_k\setminus r} y_S^k \leq p_k c_e^k, && \forall e\in E, k = 1, 2, ..., m, \\
& \sum_{S\subseteq T_k} y_S^k \leq p_k h_k(T_k), && \forall T_k\subseteq D_k\setminus r, k = 1, 2, ..., m, \\
& y_S^k \geq 0, && \forall (S,k)\in\mathcal{C}.
\end{aligned}
$$

In the above dual formulation, the variable y_S^k can be interpreted as the budget of client-scenario pair (S, k).

2.2. **The primal-dual algorithm.** Motivated by the algorithm for the deterministic prize-collecting Steiner tree problem [17], we propose an approximation algorithm to solve the SPCSTP based on the primal-dual technique. In this algorithm, we construct a feasible solution to the original problem through a dual feasible solution, instead of directly solving relaxed linear program and applying rounding techniques to obtain a feasible solution to the original problem.

We briefly introduce the overall framework of the algorithm before presenting the details. Step 0 is to initialize the algorithm: all the dual variables are zero, all the clients (terminal vertices) are not punished and all the edges are not purchased. In step 1, the algorithm is devoted to constructing a dual feasible solution. In the last step, the algorithm constructs the feasible solution of the original problem based on the dual feasible solution obtained in Step 1.

In order to understand the algorithm better, we present some intuitive explanations for Step 1 as follows. In Step 1.1, we define the property of components including active and inactive. Step 1.2 corresponds to three cases. Case 1 corresponds to purchasing a new edge in the first stage. Case 2 corresponds to purchasing a new edge in the second stage. Case 3 corresponds to the client subset which is punished.

The details of our algorithm are as follows:

Algorithm 1: Primal-dual algorithm

Step 0. Initialization
$y_S^k := 0$, $\widehat{T}_k := \emptyset$ $(k = 1, 2..., m)$, $\overline{E}_0 := \emptyset$, $\overline{E}_k := \emptyset$ $(k = 1, 2..., m)$, $\mathcal{C}_o := \{\{v, k\} | \forall (v, k) \in \mathcal{C}\}$, where $\mathcal{C}_o$ denotes all component sets.
Step 1. (Constructing a dual feasible solution.)

Step 1.1. For each component $C \in \mathcal{C}_o$, let us define $\lambda(C) = 1$ if $\exists\ k \in \{1, 2, ...m\}$ s.t. $C \subseteq T_k \subseteq D_k \setminus r$ and $\sum_{S \subseteq T_k} y_S^k < P_k h_k(T_k)$; otherwise, $\lambda(C) = 0$. If $\lambda(C) = 1$, C is active; if $\lambda(C) = 0$, C is inactive.

Step 1.2. Increase these y_S^k uniformly until one of the following three events happens:

Case 1. There exists $C_p, C_q \in \mathcal{C}_o$ such that the first cost constraint of an edge $e = (i, j)$ becomes tight where $i \in C_p$ and $j \in C_q$.

Case 2. There exist $C_{p'}, C_{q'} \in \mathcal{C}_o$ such that the second cost constraint of an edge $e = (i, j)$ becomes tight where $i \in C_{p'}$ and $j \in C_{q'}$.

Case 3. There exists $k \in \{1, 2, ..., m\}$ such that the third penalty constraint of a vertex set T_k becomes tight.

- If Case 1 occurs, update $\overline{E}_0 = \overline{E}_0 \bigcup \{e\}$ and $\mathcal{C}_o = \mathcal{C}_o \bigcup \{C_{pq}\} - \{C_p\} - \{C_q\}$ in which $C_{pq} = C_p \bigcup C_q \bigcup \{e\}$. If $r \in \{C_{pq}\}$ then $\lambda(C_{pq}) \leftarrow 0$ else $\lambda(C_{pq}) \leftarrow 1$, go to Step 1.3.
- If Case 2 occurs, update $\overline{E}_k = \overline{E}_k \bigcup \{e\}$ and $\mathcal{C}_o = \mathcal{C}_o \bigcup \{C_{p'q'}\} - \{C_{p'}\} - \{C_{q'}\}$ where $C_{p'q'} = C_{p'} \bigcup C_{q'} \bigcup \{e\}$. If $r \in \{C_{p'q'}\}$ then $\lambda(C_{p'q'}) \leftarrow 0$ else $\lambda(C_{p'q'}) \leftarrow 1$, go to Step 1.3.
- If Case 3 occurs, there exists $k \in \{1, 2, ..., m\}$ such that the penalty constraint of a vertex set $T_k \subseteq D_k \setminus r$ becomes tight and go to Step 1.3.
- If multiple cases happen simultaneously, execute one of three cases arbitrarily.

Step 1.3. Update $\lambda(C)$ for all $C \in \mathcal{C}_o$ as in Step 1.1. If there exists $C \in \mathcal{C}_o$ such that $\lambda(C) = 1$, go to Step 1.2; otherwise, stop Step 1.

Step 2. (Constructing a primal integral feasible solution.)

Consider each $e \in \overline{E}_0 \bigcup \overline{E}_k$. If removing e does not impact the connectivity of the resulting graph, remove e from the resulting graph and update $\overline{E}_0$ and $\overline{E}_k$ for each $k = 0, 1, 2, ..., m$. We choose edge set $\widehat{E}_0$ from $\overline{E}_0$ to purchase in the first stage. Then choose edge $\widehat{E}_k$ from $\overline{E}_k$ to purchase for each $k = 1, 2, ..., m$ in the second stage.

Let $\widehat{T}_k$ be the set of unserved clients in scenario k for each $k = 1, 2, ..., m$.

It is worth noting that the corresponding penalty constraint must be tight for any $T_k \subseteq D_k \setminus r$ which is not connected by the resulting graph.

3. **The analysis of the algorithm.** From Algorithm 1, we obtain a feasible solution of the SPCSTP, which is denoted by SOL. Let $cost(SOL)$ denote the total cost of SOL, i.e.,

$$cost(SOL) = \sum_{e \in \widehat{E}_0} c_e^0 + \sum_{k=1}^{m} \sum_{e \in \widehat{E}_k} p_k c_e^k + \sum_{k=1}^{m} p_k h_k(\widehat{T}_k).$$

Let OPT denote the optimal value of the SPCSTP, then we propose the main result of this paper:

Theorem 3.1. *Algorithm 1 is a 3-approximation algorithm for the SPCSTP.*

Before presenting the detailed proof, we briefly explain the main proof ideas. Dividing the cost into two parts, then we will prove the following two inequalities hold respectively:

$$\sum_{k=1}^{m} p_k h_k(\widehat{T}_k) \leq OPT, \tag{1}$$

$$\sum_{e \in \widehat{E}_0} c_e^0 + \sum_{k=1}^{m} \sum_{e \in \widehat{E}_k} p_k c_e^k \leq 2OPT. \tag{2}$$

In order to prove the inequality (1) and inequality (2) to be correct, we present the following lemmas and their detailed proof respectively.

Lemma 3.2. *Let y_S^k denote the dual solution obtained from Algorithm 1, from our algorithm, the penalty constraint is tight for any T_k, where $k = 1, 2, ..., m$, then for each $\widehat{T}_k$, we have*

$$\sum_{S \subseteq \widehat{T}_k} y_S^k = p_k h_k(\widehat{T}_k), where\ k = 1, 2, ..., m.$$

Proof. According to the construction of $\widehat{E}_0$ and $\widehat{E}_k$, each vertex which is not spanned by $\widehat{E}_0$ and $\widehat{E}_k$ (i.e. the vertices in some $\widehat{T}_k$ for $k = 1, 2, ..., m$) lies in some component deactivated at some point in the algorithm. Moreover, if such vertex was in some deactivated component C, then none of the vertices of C are spanned by $\widehat{E}_0$ and $\widehat{E}_k$.

Based on these observations, we can partition the vertices of $\widehat{T}_k$ into disjoint deactivated components $C_{k_1}, C_{k_2}, ..., C_{k_l}$ for some k. These are the maximal vertex set partition in $\widehat{T}_k$. Since each C_{k_j} is a deactivated component, then

$$\sum_{S \in C_{k_j}} y_S^k = p_k h_k(C_{k_j}),$$

and thus

$$\begin{aligned}\sum_{k=1}^{m} p_k h_k(\widehat{T}_k) &= \sum_{k=1}^{m}\sum_{j=1}^{l} p_k h_k(C_{k_j}) \\ &= \sum_{k=1}^{m}\sum_{j=1}^{l}\sum_{S\subseteq C_{k_j}} y_S^k \\ &\le \sum_{k=1}^{m}\sum_{S\subseteq \widehat{T}_k} y_S^k \\ &\le \sum_{k=1}^{m}\sum_{S\subseteq D_k\setminus r} y_S^k \le OPT\end{aligned}$$

which concludes the proof of inequality (1). □

Next, we will prove inequality (2). Recall that the incurred cost constraint is tight for each $e \in \widehat{E}_0$, thus the following two equalities hold:

$$\begin{aligned} c_e^0 &= \sum_{(S,k)} y_S^k, \\ \sum_{e\in\widehat{E}_0} c_e^0 &= \sum_{e\in\widehat{E}_0}\sum_{(S,k)} y_S^k = \sum_{(S,k)} |\widehat{E}_0 \bigcap \delta(S)| y_S^k. \end{aligned}$$

Analogously, the incurred cost constraint is also tight for each $e \in \widehat{E}_k$. Thus we obtain the following two equalities:

$$\begin{aligned} p_k c_e^k &= \sum_{S\subseteq D_k\setminus r} y_S^k, \\ \sum_{e\in\widehat{E}_k} p_k c_e^k &= \sum_{e\in\widehat{E}_k}\sum_{S\subseteq D_k\setminus r} y_S^k = \sum_{S\subseteq D_k\setminus r} |\widehat{E}_k \bigcap \delta(S)| y_S^k. \end{aligned}$$

Therefore, it can be concluded that above inequality (2) is equivalent to the following Lemma 3.3.

Lemma 3.3. *The dual feasible solution y_S^k satisfies the following inequality*

$$\sum_{(S,k)} |\widehat{E}_0 \bigcap \delta(S)| y_S^k + \sum_{k=1}^{m}\sum_{S\subseteq D_k\setminus r} |\widehat{E}_k \bigcap \delta(S)| y_S^k \le 2 \sum_{(S,k)\in\mathcal{C}} y_S^k. \tag{3}$$

First we present the framework to prove the above lemma by induction on the main loop. At the beginning of Algorithm 1, all the dual variables $y_S^k = 0$ for any (S,k) $(k = 0, 1, 2, ..., m)$. Obviously the inequality (3) holds. Thus if we can prove the incremental cost of the left side is bounded by the incremental cost of the right side in every iteration, Lemma 3.3 will be proved.

The details of the proof are as follows. Let $\mathcal{C}_o$ denote the set of all components at the beginning of each iteration. Furthermore we use $\mathcal{C}_o^{active}$ and $\mathcal{C}_o^{inactive}$ to denote the sets of active components and inactive components in $\mathcal{C}_o$, respectively. In fact, only the dual variables of all the active components can be increased in each iteration.

For every active component C belonging to $\mathcal{C}_o^{active}$, the algorithm raises the corresponding dual variable y_S^k. Suppose that the growth rate of such dual variable is ϵ. Then the left side of inequality (3) increases by

$$\sum_{C\in\mathcal{C}_o^{active}} |\widehat{E}_0 \bigcap \delta(C)|\epsilon + \sum_{k=1}^{m}\sum_{C\in\mathcal{C}_o^{active}} |\widehat{E}_k \bigcap \delta(C)|\epsilon,$$

and the right side of inequality (3) will increase by

$$2|\mathcal{C}_o^{active}|\epsilon.$$

Below we prove that

$$\sum_{C\in\mathcal{C}_o^{active}} |\widehat{E}_0 \bigcap \delta(C)| + \sum_{k=1}^{m}\sum_{C\in\mathcal{C}_o^{active}} |\widehat{E}_k \bigcap \delta(C)| \leq 2|\mathcal{C}_o^{active}|. \tag{4}$$

It is not straightforward to prove the inequality (4) directly. Instead, we consider using other tools to complete the proof. Based on the structure of $\mathcal{C}_o$, we construct a new graph $H = (\overline{V}, \overline{E})$ as follows: each vertex of H corresponds to a unique component in $\mathcal{C}_o$, i.e., $\overline{V} = \{\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_{|\mathcal{C}_o|}\}$. The edge set $\overline{E}$ consists of such edges e that $e \in \widehat{E}_0 \bigcap \delta(C)$ or $e \in \widehat{E}_k \bigcap \delta(C)$ for some $C \in \mathcal{C}_o$.

The degree of vertex $\bar{v}$ in H consists of two parts: $d_0(\bar{v})$ and $d_k(\bar{v})$, i.e., $d_H(\bar{v}) = d_0(\bar{v}) + d_k(\bar{v})$. Let A denote the vertex subset of H such that each vertex in A corresponds to a unique component in $\mathcal{C}_o^{active}$, i.e., $A = \{\bar{v}_i : C_i \in \mathcal{C}_o^{active}\}$. Similarly B denote the vertex subset that corresponds to the components of $\mathcal{C}_o^{inactive}$ with nonzero degree, that is, we discard all isolated inactive vertices in H.

The left side of inequality (4) can be rewritten as $\sum_{v\in A} d_0(\bar{v}) + \sum_{k=1}^{m}\sum_{v\in A} d_k(\bar{v})$, and the right side of inequality (4) is $2\cdot|A|$. Based on the above observations, we can rewrite inequality (4) in the following term:

$$\sum_{\bar{v}\in A} d_0(\bar{v}) + \sum_{k=1}^{m}\sum_{\bar{v}\in A} d_k(\bar{v}) \leq 2\cdot|A|. \tag{5}$$

Thus it is sufficient to prove inequality (5). We present the following lemmas for this purpose.

Lemma 3.4. *The graph H is a forest.*

Proof. From the structure of the Algorithm 1, we conclude that there is no cycle in the resulting graph. Besides, the vertex set $\overline{V}$ of H can be viewed as the contracted components of $\mathcal{C}_o$. Thus H is a forest. □

Lemma 3.5. *There can be at most one inactive leaf in H, which must correspond to the component that contains r.*

Proof. Suppose that $\bar{v}$ is an inactive leaf of H and is incident with edge e, $C_{\bar{v}}$ is the inactive component corresponding to $\bar{v}$ and the root r is not contained in $C_{\bar{v}}$, then $C_{\bar{v}}$ is deactivated by Algorithm 1. According to the construction of the resulting graph, then e is not in the resulting graph, which is a contraction. Therefore the assumption is invalid. That is, there can be at most one inactive leaf in H, which must correspond to the component containing r. □

Lemma 3.6. *For every vertex $\bar{v} \in B$ except the only inactive leaf in H (if such leaf exists), we have $d_H(\bar{v}) \geq 2$.*

Proof of Lemma 3.3:

According to Lemma 3.4, H is a forest and thus it contains at most $|A|+|B|-1$ edges. Therefore, the sum of degree of H is at most $2\cdot(|A|+|B|-1)$. Then we can obtain the following inequality:

$$\sum_{\bar{v}\in A\bigcup B} d_H(\bar{v}) \leq 2(|A|+|B|-1).$$

From Lemma 3.5 and Lemma 3.6, we have

$$\sum_{\bar{v}\in B} d_H(\bar{v}) \geq 2|B|-1.$$

Therefore, we can obtain the following inequality:

$$\begin{aligned}\sum_{\bar{v}\in A} d_0(\bar{v}) + \sum_{k=1}^{m}\sum_{\bar{v}\in A} d_k(\bar{v}) &= \sum_{\bar{v}\in A\bigcup B} d_H(\bar{v}) - \sum_{v\in B} d_H(\bar{v}) \\ &\leq 2\cdot(|A|+|B|-1) - (2\cdot|B|-1) \\ &\leq 2\cdot|A|,\end{aligned}$$

i.e., the inequality (5) holds. According to the above results, it can be concluded that we have completed the proof of Lemma 3.3.

Proof of Theorem 3.1:

From the equivalence of inequality (2) and inequality (3), Lemma 3.3 implies that inequality (2) is also correct. Based on Lemma 3.2 and Lemma 3.3, we have shown that both the inequality (1) and the inequality (2) are correct, which complete of the proof of our main result; namely, Algorithm 1 is a 3-approximation algorithm.

4. **Conclusion.** In this paper, we consider the stochastic prize-collecting Steiner tree problem with polynomial scenarios and propose a 3-approximation algorithm based on the primal-dual technique. We will consider to design better approximation algorithms for the general stochastic prize-collecting Steiner tree problem, for example, the number of scenarios may be an exponential scale, which will be more challenging and interesting in the future.

Acknowledgments. This research is supported or partially supported by the National Natural Science Foundation of China(Nos.11871280,11871081,11771386, 11728104), Beijing Natural Science Foundation Project (No.Z200002),the Zhejiang Provincial Natural Science Foundation (No. LY20A010013), the Natural Sciences and Engineering Research Council of Canada (NSERC) (No.06446) and Qinglan Project.

REFERENCES

[1] A. Archer, M. Bateni, M. Hajiaghayi and H. Karloff, Improved approximation algorithms for prize-collecting Steiner tree and TSP, *SIAM Journal on Computing*, **40** (2011), 309–332.

[2] M. Bellare, S. Goldwass, C. Lund and A. Russell, Efficient probabilistically checkable proofs and applications to approximations, in *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, (1993), 294–304.

[3] P. Berman, M. Karpinski and A. Zelikovsky, 1.25-approximation algorithm for Steiner tree problem with distances 1 and 2, in *Algorithms and Data Structures*, (eds. F. Dehne, M. Gavrilova, J.R. Sack, C.D. Tóth), Springer Press. (2009), 86–97.

[4] M. Bern and P. Plassmann, The Steiner problem with edge lengths 1 and 2, *Information Processing Letters*, **32** (1989), 171–176.
[5] D. Bienstock, M. X. Goeman, D. Simchi-Levi and D. Williamson, A note on the prize collecting traveling salesman problem, *Mathematical Programming*, **59** (1993), 413–420.
[6] A. Borchers and D.-Z. Du, The k-Steiner ratio in graphs, *SIAM Journal on Computing*, **26** (1997), 857–869.
[7] G. Borradaile, P. Klein and C. Mathieu, An $O(n \log n)$ approximation scheme for steiner tree in planar graphs, *ACM Transactions on Algorithms*, **5** (2009), 1–31.
[8] J. Byrka, F. Grandoni, T. Rothvoß and L. Sanità, Steiner tree approximation via iterative randomized rounding, *Journal of the ACM*, **60** (2013), 1–33.
[9] M. Cheung, J. Mestre, D. B. Shmoys and J. Verschae, A primal-dual approximation algorithm for min-sum single-machine scheduling problems, *SIAM Journal on Discrete Mathematics*, **31** (2017), 825–838.
[10] S. Chopra and M. R. Rao, The Steiner tree problem I: Formulations, compositions and extension of facets, *Mathematical Programming*, **64** (1994), 209–229.
[11] S. Chopra and C.-Y. Tsai, Polyhedral approaches for the Steiner tree problem on graphs, *Steiner Trees in Industry*, Comb. Optim., 11, Kluwer Acad. Publ., Dordrecht, 2001, 175–201.
[12] R. Cole, R. Hariharan, M. Lewenstein and E. Porat, A faster implementation of the GoemansWilliamson clustering algorithm, in *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, (2001), 17–25.
[13] D. Du, R. Lu and D. Xu, A primal-dual approximation algorithm for the facility location problem with submodular penalties, *Algorithmica*, **63** (2012), 191–200.
[14] P. Feofiloff, C. G. Fernandes, C. E. Ferreira and J. C. de. Pina, Primal-dual approximation algorithms for the prize-collecting Steiner tree problem, *Information Processing Letters*, **103** (2007), 195–202.
[15] M. R. Garey and D. S. Johnson, The rectilinear Steiner tree problem is NP-complete, *SIAM Journal on Applied Mathematics*, **32** (1977), 826–834.
[16] E. N. Gilbert and H. O. Pollak, Steiner minimal trees, *SIAM Journal on Applied Mathematics*, **16** (1968), 1–29.
[17] M. X. Goemans and D. P. Williamson, A general approximation technique for constrained forest problems, *SIAM Journal on Computing*, **24** (1995), 296–317.
[18] A. Gupta, M. Pál, R. Ravi and A. Sinha, Sampling and cost-sharing: Approximation algorithms for stochastic optimization problems, *SIAM Journal on Computing*, **40** (2011), 1361–1401.
[19] H. Heitsch and W. Römisch, Scenario reduction algorithm in stochastic programming, *Computational Optimization and Applications*, **24** (2003), 187–206.
[20] D. S. Hochbaum, Approximation algorithm for set covering and vertex cover problems, *SIAM Journal on Computing*, **11** (1982), 555–556.
[21] K. Jain and V. V. Vazirani, Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation, *Journal of the ACM*, **48** (2001), 274–296.
[22] D. S. Johnson, M. Minkoff and S. Phillips, The prize collecting Steiner tree problem: Theory and practice, in *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, ACM, New York, (2000), 760–769.
[23] R. M. Karp, Reducibility among combinatorial problems, *Complexity of Computer Computations*, Plenum, New York, 1972, 85–103.
[24] M. Karpinski and A. Zelikovsky, New approximation algorithms for the Steiner tree problems, *Journal of Combinatorial Optimization*, **1** (1997), 47–65.
[25] H. J. Prömel and A. Steger, A new approximation algorithm for the Steiner tree problem with performance ratio 5/3, *Journal of Algorithm*, **36** (2000), 89–101.
[26] R. Ravi and A. Sinha, Hedging uncertainty: Approximation algorithms for stochastic optimization problems, *Mathematical Programming*, **108** (2006), 97–114.
[27] G. Robins and A. Zelikovsky, Tighter bounds for graph Steiner tree approximation, *SIAM Journal on Discrete Mathematics*, **19** (2005), 122–134.
[28] R. Schultz, L. Stougie and M. H. van der Vlerk, Two-stage stochastic integer programming: A survey, *Statistica Neerlandica*, **50** (1996), 404–416.
[29] D. P. Williamson, M. X. Goemans, M. Mihail and V. V. Vazirani, A primal-dual approximation algorithm for generalized steiner network problems, *Combinatorica*, **15** (1995), 435–454.

[30] A. Z. Zelikovsky, A faster 11/6 approximation algorithm for the Steiner problem in graphs, *Information Processing Letters*, **46** (1993), 79–83.

Received November 2020; revised April 2021.

E-mail address: B201806011@emails.bjut.edu.cn
E-mail address: 416754014@qq.com
E-mail address: sunyuefang@nbu.edu.cn
E-mail address: ddu@unb.ca
E-mail address: zhangxiaoyannjnu@126.com