Steiner Forests on Stochastic Metric Graphs

Vangelis Th. Paschos*
paschos@lamsade.dauphine.fr
LAMSADE, CNRS UMR 7024 and Université Paris-Dauphine, France
Orestis A. Telelis*
telelis@di.uoa.gr
Department of Informatics and Telecommunications, University of Athens, Greece
Vassilis Zissimopoulos*
vassilis@di.uoa.gr
Department of Informatics and Telecommunications, University of Athens, Greece

Abstract

We consider the problem of connecting given vertex pairs over a stochastic metric graph, each vertex of which has a probability of presence independently of all other vertices. Vertex pairs requiring connection are always present with probability 1. Our objective is to satisfy the connectivity requirements for every possibly materializable subgraph of the given metric graph, so as to optimize the expected total cost of edges used. This is a natural problem model for cost-efficient Steiner Forests on stochastic metric graphs, where uncertain availability of intermediate nodes requires fast adjustments of traffic forwarding. For this problem we allow a priori design decisions to be taken, that can be modified efficiently when an actual subgraph of the input graph materializes. We design a fast (almost linear time in the number of vertices) modification algorithm whose outcome we analyze probabilistically, and show that depending on the a priori decisions this algorithm yields 2 or 4 approximation factors of the optimum expected cost. We also show that our analysis of the algorithm is tight.

1 Introduction

We consider the problem of laying out routes that connect simultaneously given source-destination vertex pairs over a metric graph $G_0(V_0, E_0)$. Vertices of the metric graph G_0 other than the sources and destinations may be used, but we are uncertain of their availability, in that each such vertex is present with some probability independently of all other vertices. Sources and destinations are present with probability 1. Our objective is to take some a priori decisions regarding the layout of required routes, so as to be able to come up with feasible routes for every possibly materializable subgraph $G_1(V_1, E_1)$, $V_1 \subseteq V_0$, of G_0, and minimize the expected total cost of edges used over the distribution of all such subgraphs. This is the well known *Steiner Forest* problem, defined over a *stochastic* metric graph G_0.

A brute-force way to cope with this problem is to precompute a feasible and approximate (or maybe optimum) solution for every possible subgraph of G_0 that may materialize, and

*Authors partially supported by the Greek Ministry of Education and Research under the project PYTHAGORAS II

apply an appropriate solution when the subgraph actually appears. In principle there need not be a constraint on the computational effort applied for taking a priori decisions, as long as they support fast response to the actually materialized data. In this light however, we require that such a response should be of strictly lower complexity compared to the a priori computational effort. A straightforward pattern for implementing this setting is for example to compute an optimum a priori solution over G_0, and if this solution is not feasible for the materialized subgraph G_1, use a polynomial-time approximation algorithm to obtain a completely different feasible solution for G_1. On the other hand, a natural challenge is to design such an efficient response strategy (algorithm), that can be supported by polynomial-time computable a priori decisions. In this paper we design and analyze such a strategy for *repairing* an a priori polynomial-time computable feasible solution for G_0, so as to render it feasible for G_1. We show that this strategy also approximates the optimum expected cost over all materializable subgraphs G_1.

The problem model we consider finds natural application in networks, where uncertain availability of intermediate nodes requires fast adjustments of traffic forwarding. The Steiner Forest problem is a well-known NP-hard multicommodity network design problem (even in metric graphs), generalizing the Steiner tree problem, and the only known approximation algorithm yields approximation factor 2 and was analyzed in [1, 8] (see also [18]). Recent years have seen a detailed study and sensitivity analysis of this algorithm, mainly in the context of *Stochastic Network Design*, which owes its roots to *Stochastic Programming* [5, 4], where some elements of the input data set to an optimization problem are associated to a distribution describing their probability of occurence. Stochastic Programming was introduced by the seminal work of Dantzig [5] and thereafter has evolved into an independent discipline of Operations Research that handles uncertainty in optimization problems by usage of probabilities, statistics and mathematical programming (see [4] for a description of the field). We refer the reader to [11, 7] for approximation results on Stochastic Steiner Forest models and to [13, 12, 10, 6, 9] for additional recent approximation results on stochastic network design problems in general.

Our work is mostly related to the framework of *Probabilistic Combinatorial Optimization*, introduced in [2, 14], where repairing strategies as the one described previously are analyzed probabilistically, so that the expected cost of their outcome can be computed efficiently (this ensures that the problem of taking a priori decisions for a particular strategy belongs to class NPO). Several network design problems have been treated in the probabilistic combinatorial optimization framework, including minimum coloring [17], maximum independent set [16], longest path [15], and minimum spanning tree [3]. Apart from probabilistic analysis of repairing strategies, results in [17, 16] also include derivation of approximability properties.

The article is structured as follows. At first we introduce notation. In section 2 we present a repairing strategy (algorithm) and derive the expected cost of the repaired feasible solution for the actually materialized subgraph. Approximation properties of the proposed strategy with respect to a priori decisions (solutions) are analyzed in section 3. We show that our approximation results are tight in section 4, and conclude in section 5.

Notation In what follows we denote by $G_0(V_0, E_0)$ the input metric graph and let $\langle s_r, t_r \rangle$, $r = 1 \dots k$, denote the k pairs that we have to connect for the Steiner Forest instance. Each vertex $v_i \in V_0 \setminus \{s_r, t_r | r = 1 \dots k\}$ is accociated to a probability p_i of survival in the actually materialized graph $G_1(V_1, E_1)$, $V_1 \subseteq V_0$. Vertices s_r, t_r, $r = 1 \dots k$ are assumed to be always

present in G_1. G_1 emerges as the complete metric subgraph of G_0, by an independent random Bernoulli trial for each vertex $v_i \in V_0$.

We will elaborate on a feasible a priori solution that is a forest. We denote it as an edge subset $F_0 \subseteq E_0$, consisting of f_0 trees and write $F_0 = \cup_{l=1}^{f_0} T_{0,l}$. A feasible (possibly repaired) solution over the actually materialized subgraph G_1 will be a forest $F_1 = \cup_{l=1}^{f_1} T_{1,l}$, with $T_{1,l} \subseteq E_1$, $l = 1 \dots f_1$. The subset of F_0 that remains valid for G_1 is denoted with F_0' and we refer with $T_{0,l}'$ to the subset of the tree $T_{0,l}$ that remains valid in G_1. Thus it is $F_0' = \cup_{l=1}^{f_0} T_{0,l}'$. Given two vertices v_i and v_j of some tree T, with $[v_i \cdots v_j]_T$ we denote the set of edges of the unique path connecting v_i and v_j on T.

2 A repairing strategy

In this paragraph we design and analyze probabilistically a repairing algorithm for an a priori feasible solution F_0. When the subgraph G_1 materializes, the algorithm identifies the trees of F_0 that become disconnected in G_1 (due to abscense of some edges incident to missing vertices), and reconnects each tree separately by using additional edges from E_1. Clearly this procedure generates a Steiner Forest that is feasible for the originally given pairs $\langle s_r, t_r \rangle$, $r = 1 \dots k$, and ensures $f_1 = f_0$ i.e., both the a priori and repaired forests have the same number of trees. We explain the procedure followed by the algorithm for reconnecting a particular tree of F_0 that has been disconnected in G_1. This same procedure is followed for every such disconnected tree separately.

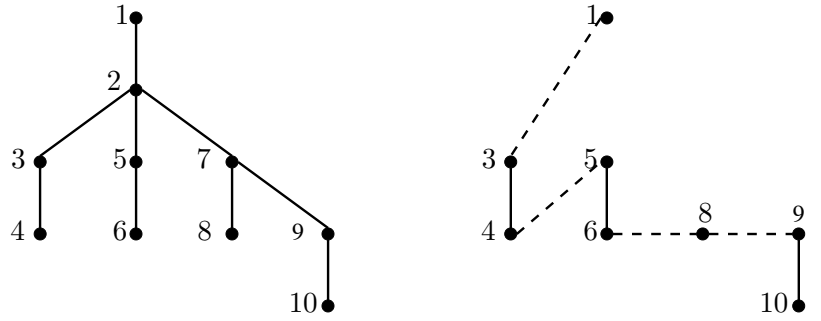
Consider the tree $T_{0,l}$, such that $T_{0,l}' \subset T_{0,l}$. The algorithm orders the vertices of $T_{0,l}$ using a Depth-First-Search, starting from an arbitrary leaf-vertex of $T_{0,l}$. Vertices of $T_{0,l}$ are inserted in an ordered list $\mathcal{L}$ in order of visitation by DFS in the following way: if v_i and v_{i+1} are two distinct vertices visited by DFS consecutively for the first time, but no (v_i, v_{i+1}) edge exists in $T_{0,l}$, then they are appended to $\mathcal{L}$ along with the parent vertex u of v_{i+1}, in the order v_i, u, v_{i+1}. Thus $\mathcal{L}$ may contain some vertices more than once (in fact, as many times as their children in $T_{0,l}$). However, $|\mathcal{L}| = O(|T_{0,l}|)$. We note that a different ordered list is produced for each tree of the a priori solution that needs to be repaired.

When the actual subgraph G_1 materializes, the algorithm sets $T_{1,l} = T_{0,l}'$. Then it removes from $\mathcal{L}$ every copy of vertex $v \in V_0 \setminus V_1$ thus producing the list $\mathcal{L}'$. It scans $\mathcal{L}'$ in order and for every two consecutive vertices v_i, v_j it inserts in $T_{1,l}$ an edge (v_i, v_j) if $i < j$ and v_i, v_j are not already connected in $T_{1,l}$. We illustrate the functionality of the repairing algorithm over a particular tree by an example.

Example Fig. 1(a) depicts a tree of an a priori solution numbered according to DFS visitation starting from a leaf-vertex. The corresponding ordered list produced in this way is $\mathcal{L} = \{1, 2, 3, 4, 2, 5, 6, 2, 7, 8, 7, 9, 10\}$. Assuming that vertices 2 and 7 are absent from the vertex set of the actually materialized subgraph, all occurences of these vertices are dropped from $\mathcal{L}$ and $\mathcal{L}' = \{1, 3, 4, 5, 6, 8, 9, 10\}$ emerges. The repairing agorithm scans $\mathcal{L}'$ in order and adds edges $(1, 3), (4, 5), (6, 8), (8, 9)$, so as to reconnect the remainders of the a priori tree, as shown in fig. 1(b).

We prove the following:

Proposition 1 *The repairing algorithm produces a connected tree $T_{1,l}$ out of tree $T_{0,l}$ of the a priori solution.*



(a) A tree of the a priori solution.

(b) The repaired tree in absence of vertices v_2, v_7. Dashed edges were added by the repairing algorithm.

Figure 1: Functionality of the repairing algorithm over a particular tree of an a priori forest.

Proof. For every vertex v_j in $\mathcal{L}'$ there is an appearance of v_j in $\mathcal{L}'$ after a vertex v_i with $i < j$, so that v_j is connected to v_i by the end of the repairing algorithm. This holds for all vertices, apart from the one appearing first in $\mathcal{L}'$. This implies that all vertices are connected into one component by the end of execution of the repairing algorithm for $T_{0,l}$. Furthermore the emerging construction cannot contain cycles for two reasons: $T_{0,l}$ did not have cycles and in order for a cycle to occur in the repaired solution $T_{1,l}$, insertion of at least one edge (v_i, v_j) is required while its endpoints have been already connected. This cannot happen by functionality of the repairing algorithm. □

Since the repairing algorithm reconnects on G_1 every single tree of the a priori solution that was disconnected, the union of all such repaired trees along with trees that survived unaffected on G_1 yields a feasible Steiner Forest on G_1. These trees remain pairwise vertex-disjoint as they were in the a priori solution, because the repairing algorithm uses only edges to reconnect trees and no such edge connects vertices belonging in different trees. Thus $f_0 = f_1$ is guaranteed.

The complexity of the repairing algorithm is almost linear in the number of vertices of G_0. Indeed, a DFS over a tree $T_{0,l}$ is of $O(|T_{0,l}|)$ time, while by using UNION-FIND disjoint sets representation for maintaining connected components during the scan of $\mathcal{L}'$, an $O(|T_{0,l}|\alpha(|T_{0,l}|))$ time is spent. Summing over all trees of the a priori forest F_0, and because $|T_{0,l}| = O(n)$, we obtain $O(n\alpha(n))$ total time for producing the final feasible forest F_1.

Theorem 1 *Given an arbitrary feasible a priori solution F_0, the expected cost of a repaired solution F_1 is:*

$$E[c(F_1)] = \sum_{l=1}^{f_1}\Big(\sum_{(v_i,v_j)\in T_{0,l}} p_i p_j c(v_i,v_j) + \sum_{(v_i,v_j)\in E(V(T_{0,l}))\setminus T_{0,l}} c(v_i,v_j) p_i p_j \times \prod_{\substack{v_l\in[v_i,v_j]_{\mathcal{L}_l}:\\ i<j\ ,\ v_i,v_j\notin[v_i,v_j]_{\mathcal{L}_l}}} (1-p_l)\Big)$$

where $V(T_{0,l})$ is the set of vertices incident to edges of $T_{0,l}$, and $E(V(T_{0,l}))$ is the set of all edges induced by vertices in $V(T_{0,l})$. Furthermore, $\mathcal{L}_l$ is the ordered list for tree $T_{0,l}$ and $[v_i, v_j]_{\mathcal{L}_l}$ the sublist of $\mathcal{L}_l$ starting at v_i and ending in v_j not including these two vertices. For all sublists not satisfying the specified restrictions we define the product to be 0.

Proof. Each individual expression summed for tree $T_{0,l}$ consists of two terms, the first one expressing the expected cost of surviving edges in the materialized subgraph (that is the expected cost of $T'_{0,l}$), while the second expresses the expected cost of edges added to $T'_{0,l}$ by the repairing algorithm, so that $T'_{0,l}$ is augmented into a feasible tree $T_{1,l}$. The first term is justified by the fact that $(v_i, v_j) \in T_{0,l}$ survives in $T'_{0,l}$ if and only if both its endpoints survive. This happens with probability $p_i p_j$, since these two events are independent.

The second term emerges by inspection of the functionality of the repairing algorithm. When G_1 materializes, missing vertices (in $V_0 \setminus V_1$) are dropped from the ordered list encoding $\mathcal{L}_l$ and the modified list $\mathcal{L}'_l$ emerges. The repairing algorithm scans $\mathcal{L}'_l$ and for every pair of consecutive vertices v_i, v_j it connects them using an edge (v_i, v_j) if and only if $i < j$ and v_i is not connected to v_j already.

Vertices $v_i, v_j \in \mathcal{L}$ both survive in $\mathcal{L}'_l$ with probability $p_i p_j$. Vertices v_i and v_j are not connected to each other if all vertices between v_i and v_j in $\mathcal{L}_l$ are missing from $\mathcal{L}'_l$, and this happens with probability $\prod_{v_l \in [v_i, v_j]_{\mathcal{L}}} (1 - p_l)$. Furthermore, neither v_i nor v_j should appear as intermediates in the sublist $[v_i, v_j]_{\mathcal{L}_l}$, otherwise they should also be missing, and would not be encountered by the repairing algorithm. Finally, the sublist $[v_i, v_j]_{\mathcal{L}_l}$ should not be empty, otherwise a surviving edge (v_i, v_j) is implied, rendering v_j connected to v_i. □

Clearly the expression given in theorem 1 is computable in polynomial-time. Thus:

Corrolary 1 *The problem of a priori optimizing the steiner forest problem on stochastic metric graphs, for the proposed repairing algorithm, belongs to the class NPO.*

The problem is NP-hard: setting all survival probabilities of vertices of G_0 equal to 1, yields a deterministic steiner forest instance. Results of section 3 imply existence of a polynomial-time 4-approximation algorithm for a priori optimization of the expression of theorem 1.

3 Approximation

In this section we carry out appropriate analysis so as to show that the Steiner Forest problem over a stochastic metric graph can be approximated efficiently within a constant factor, by the repairing algorithm. The heart of our results is the following theorem:

Theorem 2 *If F_1 is a repaired feasible solution produced by the proposed repairing algorithm for the Steiner Forest problem on a stochastic metric graph, given an a priori feasible solution F_0, then $c(F_1) \leq 2c(F_0)$.*

The proof of this result is carried out by an analysis of the algorithm over each tree $T_{0,l}$ separately. The result emerges by summing over the trees of F_1. In the following we denote by $T_{r,l}$ the subset of edges added by the repairing algorithm to $T'_{0,l}$. We prove first some lemmas that will be combined towards the proof of the theorem.

Lemma 1 *For every edge $(v_i, v_j) \in T_{r,l}$ we have $c(v_i, v_j) \leq c([v_i \dots v_j]_{T_{0,l}})$.*

Proof. Immediate by the triangle inequality holding for the cost function $c : E_0 \to \Re^+$. □

According to lemma 1 we can express the cost of the repaired tree $T_{1,l}$ as follows:

$$c(T_{1,l}) = c(T'_{0,l}) + c(T_{r,l}) \leq \sum_{e \in T'_{0,l}} c(e) + \sum_{(v_i, v_j) \in T_{r,l}} c([v_i \dots v_j]_{T_{0,l}}) \tag{1}$$

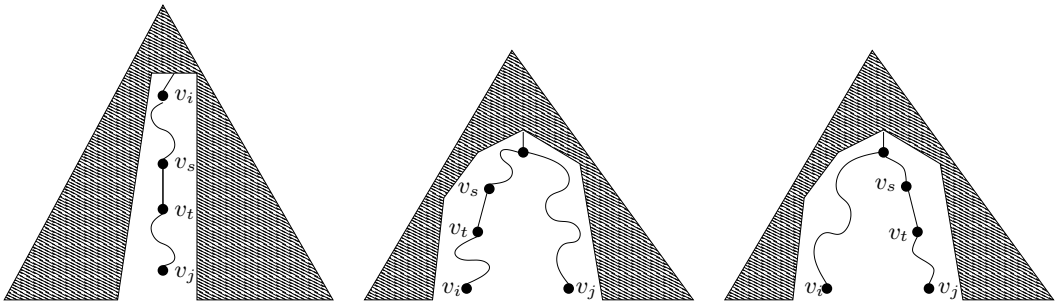


Figure 2: Three cases that may happen for edge (v_s, v_t) with respect to v_i, v_j (proof of lemma 2).

Lemma 2 *For every three distinct edges (v_i, v_j), (v_k, v_l), (v_q, v_r) in $T_{r,l}$ the paths $[v_i \cdots v_j]_{T_{0,l}}$, $[v_k \cdots v_l]_{T_{0,l}}$, $[v_q \cdots v_r]_{T_{0,l}}$, do not share an edge in common.*

Proof. By functionality of the repairing algorithm we have that $i < j$, $k < l$, $q < r$. Furthermore, if we assume without loss of generality that the vertex pairs were encountered in the order $\langle i, j\rangle$, $\langle k, l\rangle$, $\langle q, r\rangle$ during scanning of $\mathcal{L}'$, then we deduce that $j < l < r$. If the paths intersect in some common edge (v_s, v_t), then it must be $s, t \leq j$ (fig. 2 depicts all possible cases), thus $s, t < l$ and $s, t < r$. In this case edge (v_s, v_t) must have been scanned at least three times during DFS: once before visitation of each of the vertices v_j, v_l, v_r. But this contradicts the fact that a DFS scans each edge of a graph exactly twice. □

The following lemma will help us to complete the proof of the theorem:

Lemma 3 *Consider two edges (v_i, v_j), (v_k, v_l) in $T_{r,l}$. For every edge (v_s, v_t) with $(v_s, v_t) \in [v_i \cdots v_j]_{T_{0,l}} \cap [v_k \cdots v_l]_{T_{0,l}}$ it holds $(v_s, v_t) \notin T'_{0,l}$.*

Proof. The proof is by contradiction. Suppose that $(v_s, v_t) \in [v_i \cdots v_j]_{T_{0,l}} \cap [v_k \cdots v_l]_{T_{0,l}}$ and $(v_s, v_t) \in T'_{0,l}$. Without loss of generality we assume that the repairing algorithm encountered first the pair $\langle v_i, v_j\rangle$ and afterwards the pair $\langle v_k, v_l\rangle$ in $\mathcal{L}'$. It must be $i < j$, $k < l$ and $j < l$ (v_j may coincide with v_k). Since $(v_s, v_t) \in [v_i \cdots v_j]_{T_{0,l}} \cap [v_k \cdots v_l]_{T_{0,l}}$, then we must have $s, t \leq j$ and, consequently, $s, t < l$. Furthermore, it must hold either that (i) $s, t > k$ or that (ii) $s, t > i$, otherwise it should be $s, t < i$ and, given that $s, t \leq j$ also, we would deduce that (v_s, v_t) would have been scanned twice during DFS, once before visitation of v_i and once before visitation of v_j. In this case it could not have been scanned again right before visitation of v_l. Now (i) cannot hold because $k \geq j$ and $s, t < j$. If (ii) holds, i.e. $s, t > i$, it is implied that the repairing algorithm did not encounter in $\mathcal{L}'$ vertices v_k, v_l and v_i, v_j consecutively (fig. 3), which is a contradiction. □

The proof of theorem 2 can now be completed as follows:

Proof. Relation (1) can be written:

$$c(T_{1,l}) \leq \sum_{e \in T'_{0,l}} c(e) + \sum_{(v_i, v_j) \in T_{r,l}} c([v_i \ldots v_j]_{T_{0,l}}) = \sum_{e \in T'_{0,l}} c(e) + \sum_{(v_i, v_j) \in T_{r,l}} \sum_{e \in [v_i \ldots v_j]_{T_{0,l}}} c(e) \Rightarrow$$

$$c(T_{1,l}) \leq \sum_{e \in T'_{0,l}} c(e) + \sum_{(v_i, v_j) \in T_{r,l}} \Big(\sum_{e \in [v_i \ldots v_j]_{T_{0,l}} : e \in T'_{0,l}} c(e) + \sum_{e \in [v_i \ldots v_j]_{T_{0,l}} : e \notin T'_{0,l}} c(e) \Big) \qquad (2)$$

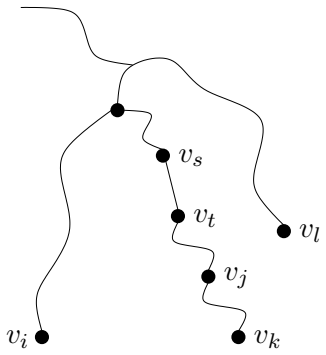


Figure 3: Case $s, t > i$ examined in the proof of lemma 3: $(v_s, v_t) \in [v_i \cdots v_j]_{T_{0,l}} \cap [v_k \cdots v_l]_{T_{0,l}}$ and suppose that $(v_s, v_t) \in T'_{0,l}$. Then obviously v_s, v_t survive in $\mathcal{L}'$ and appear as intermediates in the two pairs $\langle v_i, v_j \rangle$ and $\langle v_k, v_l \rangle$.

By lemmas 2 and 3 the following are implied:

$$\sum_{(v_i,v_j)\in T_{r,l}} \sum_{e\in[v_i \ldots v_j]_{T_{0,l}}: e\in T'_{0,l}} c(e) \leq \sum_{e\in T'_{0,l}} c(e) \tag{3}$$

$$\sum_{(v_i,v_j)\in T_{r,l}} \sum_{e\in[v_i \ldots v_j]_{T_{0,l}}: e\notin T'_{0,l}} c(e) \leq 2 \sum_{e\in T_{0,l}\setminus T'_{0,l}} c(e) \tag{4}$$

By replacing the relations (3) and (4) in the expression (2) we obtain:

$$c(T_{1,l}) \leq 2 \sum_{e\in T'_{0,l}} c(e) + 2 \sum_{e\in T_{0,l}\setminus T'_{0,l}} c(e) \leq 2c(T_{0,l}) \tag{5}$$

Summing over all trees, since $f_0 = f_1$, we obtain that $c(F_1) \leq 2c(F_0)$. □

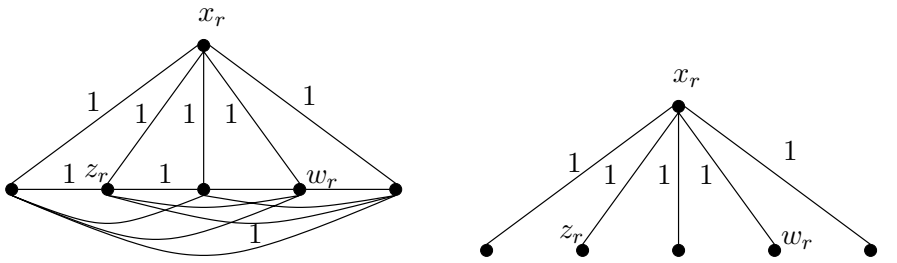
Now we can state our main approximation result:

Theorem 3 *There is an almost linear time repairing algorithm for Steiner Forest problem on stochastic metric graphs that, when applied to an α-approximate a priori feasible solution, produces feasible solutions that are 2α-approximate to the optimum expected cost.*

Proof. By theorem 2 $c(F_1) \leq 2c(F_0)$. Let $OPT(G_0)$ and $OPT(G_1)$ be the costs of an optimum Steiner forest on G_0 and G_1 respectively for the given source-destination pairs, and $c(F_0) \leq \alpha OPT(G_0)$. It is $OPT(G_0) \leq OPT(G_1)$ for every possible subgraph G_1 of G_0, because every feasible solution for G_1 is also feasible for G_0. Thus $c(F_1) \leq 2\alpha OPT(G_1)$. Taking expectation over all possible subgraphs G_1 yields $E[c(F_1)] \leq 2\alpha E[OPT(G_1)]$. □

Corrolary 2 *There is an $O(n\alpha(n))$ time repairing algorithm for the Steiner Forest problem on stochastic metric graphs, that can be supported by a polynomial-time algorithm for taking a priori decisions [1, 8], so as to yield factor 4 approximation of the optimum expected cost. The repairing algorithm is 2-approximate given an a priori optimum feasible solution.*

We note that in both cases mentioned in the corollary, the proposed repairing algorithm is faster than the algorithm used for a priori decisions, and is far more efficient than the trivial practices discussed in the introduction: in fact, any approximation algorithm used for taking a priori decisions (including the one of [1, 8]) will incur $\Omega(n^2)$ complexity.



(a) The structure used in proving tightness of analysis for some S_r, with $|S_r| = 5$. Apart from edges with cost 1, all other edges have cost 2.

(b) An optimum star centered at x_r, with cost 5.

Figure 4: Illustration of worst-case construction for showing tightness of analysis.

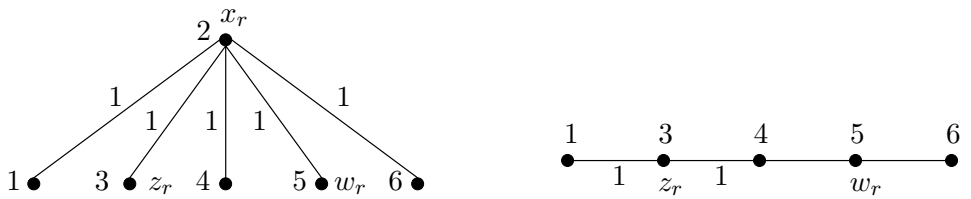
4 Tightness of Analysis

We show in this paragraph that the analysis of the repairing algorithm is tight for arbitrary a priori solution F_0. We construct a worst-case example.

We consider a metric graph $G_0(V_0, E_0)$. For some fixed constant k take k sets of vertices $S_1, \ldots, S_k$, along with a vertex $x_r \notin S_r$, $r = 1 \ldots k$, per subset. Let the input metric graph consist of the vertex set $V_0 = \left(\cup_{r=1}^{k} S_r\right) \cup \{x_r | r = 1 \ldots k\}$. We take $|S_r| = n$ so that $|V_0| = \Theta(n)$, because k is a fixed constant. We set $c(x_r, y) = 1$ for each $y \in S_r$, $r = 1 \ldots k$. For each S_r pick two distinct arbitrary vertices $w_r, z_r \in S_r$ and set $c(z_r, y) = 1$ for all $y \in S_r \setminus \{w_r, z_r\}$ and $c(z_r, w_r) = 2$. For all other edges of the graph we set their cost equal to 2. Fig. 4(a) shows the construction for a particular set S_r.

The Steiner Forest instance that we consider requires that each S_r is connected (it is trivial to express this requirement with source-destination pairs). We assume that the stochastic graph is defined by setting the survival probability of each x_r equal to p. An optimum a priori solution to this instance is defined as a forest consisting of an optimum connecting tree per vertex set S_r. We consider such an a priori solution that the corresponding tree for S_r is the star $T_r = \{(x, y) | y \in S_r\}$. Fig. 4(b) shows the construction for a particular vertex set S_r and the optimum star tree solution for this set.

Among the various cases that may occur in the actually materialized subgraph of G_0 we consider the one where all vertices x_r, $r = 1 \ldots k$ survive, and the case where all vertices x_r are missing. For the first case the a priori optimum solution remains feasible and has an optimum cost of $\sum_{r=1}^{k} |S_r|$, while in the second case, the repairing algorithm is executed for each tree T_r. Fig. 5 depicts the DFS numbering of a tree T_r by the repairing algorithm, and the corresponding repaired solution. It is easy to see that such a "chain" as the one appearing in fig. 5(b), must have a cost at least $2(|S_r| - 1) - 2 = 2(|S_r| - 2)$, because in this chain z_r may be incident to two vertices that are connected with two edges of cost 1 to it. However, the optimum cost for the materialized subgraph occurs if we connect per set S_r its vertices to z_r, and is equal to $|S_r|$. Clearly the optimum expected cost is at most $\sum_{r=1}^{k} |S_r| = k(n-1)$, while the solution produced by the repairing algorithm has an expected cost of value at least $p^k \sum_{r=1}^{k} |S_r| + 2(1 - p^k) \sum_{r=1}^{k} (|S_r| - 2) = kp^k(n-1) + 2k(1-p^k)(n-2)$. Hence, the approximation factor is asymptotically lower-bounded by: $\lim_{n\to\infty} \frac{kp^k(n-1)+2k(1-p^k)(n-2)}{k(n-1)} = p^k + 2(1-p^k)$, which approaches arbitrarily close to 2 by choosing p arbitrarily close to 0.



(a) DFS numbering produced by the repairing algorithm for star connection of S_r.

(b) A feasible tree produced over a disconnected star tree, when x_r is missing.

Figure 5: DFS numbering and repaired tree assumed in showing tightness of analysis.

5 Conclusions

We considered the Steiner Forest problem in stochastic metric graphs, where each vertex that is not a source or destination is present with a given probability independently of all other vertices. The problem amounts to coming up with a feasible Steiner Forest for every possible materializable subgraph of the given graph, so as to minimize the expected cost of the resulting solution taken over the distribution of these subgraphs. We designed an efficient algorithm that runs in almost linear time in the number of vertices that adjusts efficiently a priori taken decisions. Given that a priori decisions constitute a feasible forest on the original metric graph we were able to derive a polynomial time computable expression for the expected cost of a Steiner Forest produced by the proposed algorithm. Furthermore, we have shown that this algorithm at most doubles the cost of the a priori solution, and this leads to 2 approximation of the optimum expected cost given an optimum a priori solution, and 4 approximation given a 2 approximate solution. Our analysis of the proposed repairing algorithm was shown to be tight.

We note that for the more special case of the Steiner Tree problem in the same model, the well-known minimum spanning tree heuristic [18] that includes only vertices requiring connection, gives a feasible and 2-approximate a priori solution that trivially remains feasible and 2-approximate for the actually materialized subgraph. As a non-trivial aspect of future work we consider extending our results to the case of complete graphs with general cost functions. Simply using shortest-path distances on these graphs does not straightforwardly lead to efficient and approximate repairing algorithms.

References

[1] A. Agrawal, P. N. Klein, and R. Ravi. When Trees Collide: An Approximation Algorithm for the Generalized Steiner Problem on Networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

[2] D. Bertsimas. *Probabilistic Combinatorial Optimization*. PhD thesis, 1988.

[3] D. Bertsimas. The probabilistic minimum spanning tree problem. *Networks*, 20:245–275, 1990.

[4] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer-Verlag New York, 1997.

[5] G. W. Dantzig. Linear programming under uncertainty. *Management Science*, 1:197–206, 1951.

[6] K. Dhamdhere, R. Ravi, and M. Singh. On Two-Stage Stochastic Minimum Spanning Trees. In *Proceedings of the International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 321–334, 2005.

[7] L. Fleischer, J. Koenemann, S. Leonardi, and G. Schaefer. Simple cost sharing schemes for multicommodity rent-or-buy and stochastic steiner tree. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 663–670, 2006.

[8] M. X. Goemans and D. P. Williamson. A General Approximation Technique for Constrained Forest Problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[9] A. Gupta and M. Pál. Stochastic Steiner Trees Without a Root. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, pages 1051–1063, 2005.

[10] A. Gupta, M. Pál, R. Ravi, and A. Sinha. What About Wednesday? Approximation Algorithms for Multistage Stochastic Optimization. In *Proceedings of the International Workshop on Approximation and Randomized Algorithms (APPROX-RANDOM)*, pages 86–98, 2005.

[11] A. Gupta, Martin Pál, R. Ravi, and A. Sinha. Boosted sampling: approximation algorithms for stochastic optimization. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 417–426, 2004.

[12] A. Gupta, R. Ravi, and A. Sinha. An Edge in Time Saves Nine: LP Rounding Approximation Algorithms for Stochastic Network Design. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 218–227, 2004.

[13] N. Immorlica, D. R. Karger, M. Minkoff, and V. S. Mirrokni. On the costs and benefits of procrastination: approximation algorithms for stochastic combinatorial optimization problems. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 691–700, 2004.

[14] P. Jaillet. *Probabilistic Traveling Salesman Problems*. PhD thesis, 1985.

[15] C. Murat and V. Th. Paschos. The probabilistic longest path problem. *Networks*, 33(3):207–219, 1999.

[16] C. Murat and V. Th. Paschos. A priori optimization for the probabilistic maximum independent set problem. *Theoretical Computer Science*, 270(1–2):561–590, 2002.

[17] C. Murat and V. Th. Paschos. On the probabilistic minimum coloring and minimum k-coloring. *Discrete Applied Mathematics*, 154(3):564–586, 2006.

[18] V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Heidelberg, 2003.