Stochastic Minimum Spanning Trees and Related Problems

Pegah Kamousi Subhash Suri

Department of Computer Science
University of California
Santa Barbara, CA 93106, USA

Abstract

We investigate the computational complexity of minimum spanning trees and maximum flows in a simple model of *stochastic networks*, where each node or edge of an undirected master graph can fail with an independent and arbitrary probability. We show that computing the expected length of the MST or the value of the max-flow is $\#P$-Hard, but that for the MST it can be approximated within $O(\log n)$ factor for metric graphs. The hardness proof for the MST applies even to Euclidean graphs in 3 dimensions. We also show that the tail bounds for the MST cannot be approximated in general to any multiplicative factor unless P = NP. This stochastic MST problem was mentioned but left unanswered by Bertsimas, Jaillet and Odoni [Operations Research, 1990] in their work on a priori optimization. More generally, we also consider the complexity of linear programming under probabilistic constraints, and show it to be $\#P$-Hard. If the linear program has a constant number of variables, then it can be solved exactly in polynomial time. For general dimensions, we give a randomized algorithm for approximating the probability of LP feasibility.

1 Introduction

Uncertainty is a fact of life whether we are dealing with physical or natural systems: devices fail, bonds break, messages get corrupted, demands and supplies behave unpredictably and so on. In computer science and mathematical optimization, a number of models and approaches have been considered to account for and analyze the effect of uncertainty: random graphs [4], stochastic geometry [5], Bayesian analysis [23] and multi-stage stochastic optimization [17] being some prominent examples. In this paper, we revisit the minimum spanning tree and some related problems under a basic and natural model of uncertainty.

Suppose we are given a complete, weighted undirected graph $G = (V, E)$, on n nodes and m edges, called the *master graph*, where each node v_i is *active* (or present) with independent probability p_i. When a node is inactive, all of its incident edges are also absent. We wish to compute the *expected minimum spanning tree* cost for this stochastic network G, namely, $\sum p(H)MST(H)$, where the sum is over all node-induced subgraphs H of G, $p(H)$ is the probability with which H appears, and $MST(H)$ is the cost of its minimum spanning tree. Similarly, we may consider the stochastic version of the *max-flow* problem, where each edge $e_i \in E$ fails with probability p_i, and we seek the *expected value of the maximum flow* from a source node s to a sink node t.

Besides being fundamental estimation problems in their own right, these types of questions also arise naturally in studies of real networks where uncertainty is an inherent part of data. For instance, users or *demand* nodes in a large network appear and remain active for varying periods of time. Our stochastic MST problem models the cost of spanning these nodes in an overlay network, such as distributed multi-person game or data dissemination service. While the independent probability model of stochasticity is not new, the kind of *estimation* problems we consider appear not to have been studied. For instance, much of the current research on stochastic optimization is focused on two-stage models, where one trades the current uncertainty with future price inflation [10, 17]. On the other hand, the classical random models focus on asymptotic results on random point sets or random graphs [1]. Similarly, previously studied models of stochastic linear programming have focused on random variables in constraints (random coefficients) or in the objective function (random additive terms) with known probability distributions [20]. Our model has a more combinatorial flavor because of the binary nature of each constraint: it either applies (with probability p_i) or does not (with probability $1-p_i$). In many management decision systems, this type of binary and independent form or constraints seem very natural. More generally, we are interested in expectations over *worst-case* graphs or linear programs with *worst-case* probability distributions.

1.1 Our Results

We show that the problem of computing the expected cost of the minimum spanning tree is $\#P$-hard in general stochastic graphs where nodes are active with arbitrary probabilities. We also show that the tail bounds of the MST distribution cannot be approximated to any multiplicative factor unless $P = NP$. On the positive side, we show that if the costs in the graph are *metric*, then the expected cost of the MST can be approximated to $O(\log n)$ factor in polynomial time. We also show that the stochastic max-flow problem is $\#P$-Hard when the nodes are deterministic but the *edges* of the graph fail with arbitrary probabilities.

We then consider linear programming where each constraint c_i, namely, $a_i x \leq b_i$, is active with probability p_i. This is equivalent to a stochastic input $H = \{h_1, h_2, \ldots, h_n\}$ of halfspaces, where h_i is present (active) with probability p_i, and we may want to know the probability that the common intersection $\bigcap h_i$ is non-empty (the linear program is feasible), or to optimize the expected value of some linear objective. We show that for arbitrary dimensions, the stochastic linear

programming problem is $\#P$-Hard, by reduction from the stochastic max-flow problem. We then present a polynomial-time approximation scheme to estimate the probability that the linear program is feasible. When the dimension d is fixed, the stochastic linear programming can be solved easily in $O(n^{d+1})$ time.

1.2 Related Work

A vast literature exists on combinatorial structures under probabilistic distributions, but much of it has a different flavor than ours. In the following, we mention a few prominent models and techniques that most directly relate to our research. The seminal work of Erdös and Rényi [12] has led to a significant body of work on properties of random graphs. A major theme of this research is to estimate threshold values of probability parameters at which some fundamental properties (asymptotically) emerge or disappear in random graphs–see [6] for more details. Stochastic geometry (also called geometric probability) deals with properties of, and probabilistic analysis of algorithm for, points drawn at random form a geometric space. Classical results include formulae for the expected length of the minimum spanning tree, traveling salesman tour, node degree etc. for n random points drawn from a unit cube, or ball—see results of Beardwood, Halton, Hammersley [3], Karp [21], Frieze [14, 15] and Steele [28]. Unlike the random instances studied in these models, we consider *worst-case* instances with *worst-case* probabilities associated with each element. Another line of research in combinatorial optimization under uncertainty which is currently very active is Multi-Stage Stochastic Optimization [24, 29]. This model trades off current uncertainty in demand with future inflation in prices. Typically, only part of the input is revealed in the first stage. The rest of the input is revealed in the second stage when resources are more expensive. There is a rich body of work in this area, focusing on combinatorial algorithms as well as multi-stage linear programming [10, 13, 17, 22]. The 2-stage optimization work is different from ours both in the type of questions asked and input to the problem in having an additional first (deterministic) stage.

One model which has more similarity to our model is *a priori* optimization, inspired by the observation that many optimization problems are solved repeatedly for instances drawn from a common master instance. Bertsimas [4] and Jaillet [18] proposed creating a single master solution which is adapted to individual instances by a computationally simple heuristic, where the goal is to minimize the expected cost of the adapted master solution over all probabilistic instances. By contrast, we wish to estimate the optimal cost (over all instances) *without* the restriction of using a single master solution. In fact, the problem of estimating the cost of the MST without the a priori constraint was mentioned in [4] but left unanswered.

A problem which fits well to our model is the *Network Reliability Problem*. In [27], Rosenthal considers the problem of estimating the probability that a network is connected, given (rational) failure probabilities of its elements (edges and nodes), and proves it to be NP-hard –Later, Provan and Ball [26] show that the network reliability problem is in fact $\#P$-hard. Curiously, the problem of estimating the cost of the stochastic MST, a fundamental combinatorial structure, has not been explored, either in the graphical or in the geometric setting. In the next section, we consider the complexity of this problem for general graphs.

2 Complexity of Stochastic MST in General Graphs

Suppose we are given a weighted, undirected complete *master graph* $G = (V, E)$, on n nodes and m edges. Each node v_i of the master graph is *active* (or present) with independent probability p_i, and inactive (or dead) otherwise. When a node is inactive all of its incident edges are also absent. The

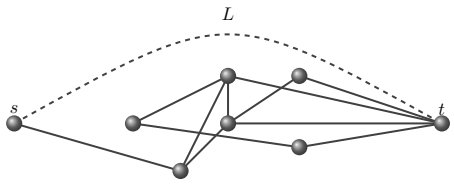


Figure 1: Illustrating the construction for the proof of Theorem 2.1.

expected cost of the minimum spanning tree of this stochastic graph G is defined as $\sum p(H)MST(H)$, where the sum is over all node-induced subgraphs H of G, $p(H)$ is the probability of H being active, and $MST(H)$ is the cost of its minimum spanning tree.

In this section, we show that even when node probabilities are restricted to the set $\{\frac{1}{2}, 1\}$, the problem of computing the expected MST cost is $\#P$-hard. In particular, we prove the following result.

Theorem 2.1. *The problem of computing the expected cost of the minimum spanning tree for a complete undirected graph where each node v_i is active with independent rational probability p_i, for $0 \leq p_i \leq 1$, is $\#P$-hard.*

We prove this result by a reduction from the classical *S-T Node Connectedness* problem, which was shown to be $\#P$-hard by Valiant [30]. The *S-T* node-connectedness problem is, given an undirected graph $G = (V, E)$ and a pair of nodes s and t, compute the *number* of subsets of V whose removal leaves s and t path-connected. Given an instance of the *S-T* node connectedness problem, we construct a stochastic graph instance $G' = (V', E')$ so that from the expected cost of the MST of G', in polynomial time we can compute the number of subsets of V whose removal leaves s and t connected. The graph G' is constructed as follows, and illustrated in Figure 1. Let $G = (V, E)$ be an instance of the *S-T* node connectedness problem. First, if G contains the edge (s, t), then the connectedness problem is trivial: no subset of $V - \{s, t\}$ can disconnect s and t, so we assume w.l.o.g. that $(s, t) \notin E$. In our stochastic graph G', we set $V' = V$, and include each edge $(u, v) \in E$ also in E' *with cost* 0. Next, we add a new edge (s, t) between s and t with cost equal to L, for $L > 0$ to be chosen later. We then complete the graph G' by adding all those edges that are not present in E, and set the cost of each of these to an arbitrary value $L' \gg L$. (The edges with weight $L' \gg L$ are added only to ensure that each induced subgraph is always connected and the expected cost of the MST is a finite number.)

Finally, we set the node probabilities as follows: the nodes s and t appear with probability 1, while each of the remaining nodes appears with probability 1/2. A simple consequence of this uniform assignment of probabilities is that each subset of $V = V'$, with a non-zero probability of occurrence, appears with equal probability. The following lemma will be a key to proving Theorem 2.1. The proof can be found in the appendix.

Lemma 2.2. *The number of times the edge (s, t) occurs in the MST over all subset of V' is exactly the same as the number of subsets of V that leave s and t* disconnected *in G.*

We are now ready to complete the proof of Theorem 2.1.

Proof of Theorem 2.1. Constructing G' from an instance G of the node-connectedness problem clearly takes polynomial time. We now compute the expected MST cost of G' under two different values of L, the cost of the edge (s, t), once with $L = n$ and once with $L = n + 1$; these choices are for convenience only, and many other choices will work as well. Let $\mathbb{E}[MST_1]$ and $\mathbb{E}[MST_2]$ denote the

expected values for these two instances, respectively. For each pair of nodes i, j in V, let w_{ij} denote the cost (weight) of the edge (i, j) in G' and let p_{ij} be the probability that the edge (i, j) is used in the MST. From the linearity of expectation we have

$$\mathbb{E}[MST_1] = \sum_{i,j \in V} p_{ij} w_{ij} = \sum_{(i,j) \in V \times V, (i,j) \neq (s,t)} p_{ij}.w_{ij} + p_{st}.n.$$

Furthermore, as we increase the weight of (s, t), while making sure that it remains smaller than L', the probabilities p_{ij} remain the same for all i, j, and only the cost of a single edge (s, t) increases. Therefore, we also have

$$\mathbb{E}[MST_2] = \sum_{i,j \in V} p_{ij} w_{ij} = \sum_{(i,j) \in V \times V, (i,j) \neq (s,t)} p_{ij}.w_{ij} + p_{st}.(n+1).$$

¿From these equalities, we get that $\mathbb{E}[MST_2] - \mathbb{E}[MST_1] = p_{st}$. Thus, the number of times the edge (s, t) appears in all the MSTs over all the active subsets of G' is precisely $2^{n-2}(\mathbb{E}[MST_2] - \mathbb{E}[MST_1])$, which is also the number of subsets of $V - \{s, t\}$ that disconnect s and t in G. This proves that computing the expected MST cost is $\#P$-hard. □

Next, we consider the problem of approximating the tail bounds for the distribution of the minimum spanning tree cost.

2.1 Hardness of Tail bound Approximation for the Stochastic MST

We prove that one cannot hope to approximate the tail bounds of the distribution of the stochastic minimum spanning tree, within any factor in polynomial time.

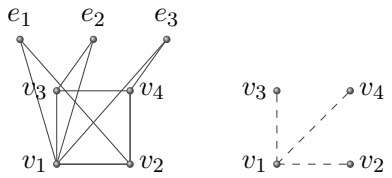
Theorem 2.3. *Given a graph $G = (V, E)$ and a rational failure probability p_i for each $v_i \in V$, it is NP-hard to approximate, within any factor, the tail bounds for the expected weight of the minimum spanning tree of G. In other words, given $L > 0$, it is NP-Hard to compute $\hat{r} > 0$ such that*

$$\frac{1}{\alpha} \Pr[\mathbb{E}(MST) \leq L] \leq \hat{r} \leq \alpha \Pr[\mathbb{E}(MST) \leq L]$$

The reduction is from the minimum vertex cover problem. Given a graph $G = (V, E)$, for which we ask for the size of the minimum vertex cover, we construct a graph $G' = (V', E')$. G' contains a set S_v of $|V|$ nodes, associated with the nodes of G, and a set S_e of $|E|$ nodes, associated with the edges of G. All the nodes in S_v are connected to each other with edges of weight 0. Suppose the node $v_i \in S_e$ is associated with the edge $e_i = (u, v)$ in G. Then v_i is connected to the two nodes of S_v associated with u and v with edges of weight 0. It will then be connected to all the other nodes in S_v with edges of weight $L > 0$. Figure 2.1 shows the construction. Only the edges of weight zero are shown.

We let each node in S_v be present with probability p (to be defined later), while the rest of the nodes are present with probability 1. We are interested in the tail bounds for the minimum spanning tree problem on this instance, i.e., for a given number ℓ, we ask for the probability that the expected weight of the minimum spanning tree is less (or similarly more) than ℓ. The following lemma is a key to our reduction. The proof can be found in the appendix.

Lemma 2.4. *Let $H \subseteq V'$ be the surviving subset of nodes, and let $MST(H)$ be the weight of the minimum spanning tree of H. Then $MST(H) = 0$ if and only if the subset of S_v nodes that survive, forms a vertex cover in the original graph, G.*



$(a)G'$ (b) The original graph, G

Figure 2: The construction

Suppose we can approximate the tail bounds for the MST problem up to a multiplicative factor α, i.e., we output a number $\hat{r}$ such that

$$1/\alpha Pr[MST(G) < L] \leq \hat{r} \leq \alpha Pr[MST(G) < L].$$

Let $p = 1/(\alpha^2 2^{|V|})$. If there exists a vertex cover VC of size K in G, then the probability that the minimum spanning tree of G' is lighter than L is at least the probability that the set of S_v nodes associated with VC is up, i.e.,

$$Pr[MST(G) < L] \geq Pr[VC \text{ is up }] \geq p^K,$$

and therefore $\hat{r} \geq p^K/\alpha$. If there is no vertex cover of size K, then for the MST to be lighter than L, at least $K+1$ nodes from S_v should be up, and therefore

$$Pr[MST(G) < L] \leq Pr[K+1 \text{ or more nodes are up }] < 2^{|V|} p^{K+1} = p^k/\alpha^2.$$

In this case, $\hat{r} < p^K/\alpha$, and we can therefore distinguish between the two cases.

In the next section, we will show that the stochastic MST problem remains hard, even for Euclidean graphs in any dimension $d \geq 3$.

3 Hardness of the Stochastic MST in R^3

Theorem 3.1. *Computing $\mathbb{E}[MST(S)]$ is #P-Hard for a stochastic set S of n points in d-dimensional Euclidean space, for $d > 2$.*

Our hardness proof uses a reduction from the problem of *2-terminal network reliability* (2-NRP), which is known to be #P-Hard [26, 27]. In the 2-terminal network reliability, we are given an undirected graph $G = (V, E)$, two special nodes s and t, and a rational failure probability for each edge $e_i \in E$. The goal is to compute the probability that s and t are connected in this stochastic graph. We now describe our reduction from this problem to the stochastic geometric MST.

3.1 The Construction

Figure 3(a) shows our construction. We first (arbitrarily) order the nodes and edges of G, so that the nodes are numbered 1 through $|V|$, the edges are numbered 1 through $|E|$, and the nodes s and t are adjacent in the ordering. Corresponding to each node $v_i \in V$, we create a horizontal line $y = 2i$ with $2|E|$ points placed uniformly on it, with inter-point distance 1. We call this the *virtual node*

of v_i. Thus, all the points associated with a virtual node can be connected using edges of length ≤ 1, which we call *short edges*. This creates a $|V| \times 2|E|$ grid of points in the plane, in which the horizontal lines corresponding to virtual nodes are separated by distance 2. For all these points, we set the probability $p_j = 1$.

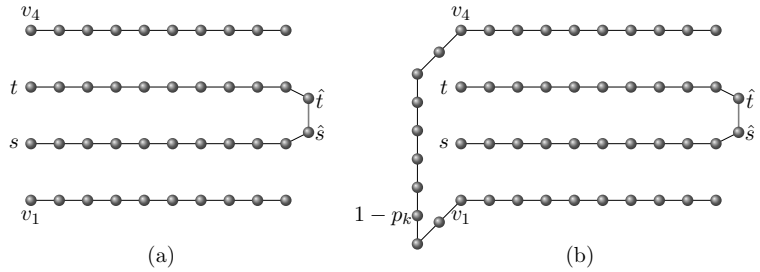


Figure 3: (a) The construction of points, and the special points $\hat{s}$ and $\hat{t}$. (b) A sample short path.

We add two additional points $\hat{s}$ and $\hat{t}$ to the end of the lines associated with s and t in such a way that $\hat{s}$ and $\hat{t}$ are at distance 1 from their predecessors on the line, and the distance between $\hat{s}$ and $\hat{t}$ is ℓ, for some value $1 < \ell < 1.5$. (This can be done easily and any choice of ℓ in this range is valid).

Next, we encode the edges of G into paths of short (length ≤ 1) edges in our geometric construction. In particular, suppose the edge $e_k \in E$ in the input graph G joins nodes v_i and v_j, and that it has failure probability p_k. We create a sequence of points that forms a path of short edges, called a *virtual edge*, in the 3rd dimension joining the $(2k-1)$st point of the virtual node v_i to the $(2k-1)$st point of the virtual node v_j. Specifically, the virtual edge corresponding to e_k is created by introducing points at locations $(2k-1, 2i, 1)$, $(2k-1, 2i, 2)$, $(2k-1, 2j, 1)$, $(2k-1, 2j, 2)$, and point $(2k-1, l, 2)$ for $l = 2i, 2i+1, \ldots, 2j$ (assuming that $i < j$). We designate one of these points $(2k-1, 2i+1, 2)$ as *representative* of this virtual edge, and denote it as r_k. All points on the virtual edge corresponding to e_k appear with probability 1 *except* the representative point r_k, which appears with probability $(1 - p_k)$, namely the *survival* probability of edge e_k in G. This is our final set of points S. Figure 3(b) shows the construction for an example edge $e_k = (v_1, v_4)$. We observe that the virtual edge corresponding to an edge in G can be realized as a path consisting of unit-length edges connecting the virtual nodes corresponding to the endpoints of that edge.

3.2 The Analysis

The following lemma is the key ingredient of our reduction, and its proofs is given in the appendix.

Lemma 3.2. *Let G be an instance of the 2-terminal network reliability problem, and let H be a subgraph of G consisting of the edges that have survived. Let $S(H) \subseteq S$ be the subset of points in our construction excluding the representatives corresponding to the failed edges of G. Then, s and t are connected in H if and only if the segment $\hat{st}$ is not included in the MST of the set $S(H)$.*

We now show how to compute the probability of $\hat{st}$ being included in the MST given the expected length of the MST of S. For any two points $a, b \in S$, we use the *indicator function* $I(a, b)$ to denote when the line segment ab is included in the MST. Furthermore, let $p(a, b)$ denote the probability of the segment ab being in MST, and let $\ell(a, b)$ be the Euclidean length of the segment ab. From the linearity of expectation, we have

$$\mathbb{E}(MST) = \sum_{a,b \in S} p(a,b)\ell(a,b)$$

Next, suppose we increase the length of the segment $\hat{st}$ by 0.1 by simply moving the two points apart slightly while keeping their distances from their predecessors on their respective lines equal to one, as before. Let $\mathbb{E}[MST_1]$ be the expected weight of the minimum spanning tree with the original length of $\hat{st}$, and $\mathbb{E}[MST_2]$ the expected length with the new length. We have the following lemma, whose proof can be found in the appendix.

Lemma 3.3. *For all pairs of points $a, b, \in S$, $a \neq \hat{s}, b \neq \hat{t}$, the probability $p(a,b)$ of inclusion in MST_1 and MST_2 is unchanged, and $\hat{st}$ is the only edge in the MST whose weight changes.*

¿From the preceding lemma, we conclude that $\mathbb{E}[MST_2] - \mathbb{E}[MST_1] = 0.1\mathbb{E}[I(\hat{s},\hat{t})] = 0.1p(\hat{s},\hat{t})$. Finally, by Lemma 3.2, the probability of s and t being connected in G is equal to $1 - p(\hat{s},\hat{t})$, which can be calculated from the difference $\mathbb{E}(MST_2) - \mathbb{E}(MST_1)$. Thus, by running the algorithm twice with different lengths of $\hat{st}$, we get that

$$Pr[s,t \text{ connected }] = 1 - 10 * (\mathbb{E}(MST_2) - \mathbb{E}(MST_1)).$$

The reduction is clearly polynomial time since the size of the point set S is at most $O(|V|.|E|)$: there are $O(|V|.|E|)$ points on the plane, and another $O(|V|.|E|)$ points in the third dimension. This completes the proof of Theorem 3.1.

4 Stochastic MSTs in Metric Spaces

We show that a rather simple and deterministic algorithm achieves a $O(\log n)$ factor approximation for the complete graph defined over a set S of points in a metric space. The algorithm works as follows:

- Arbitrarily order the points of S as 1 through n.
- Process the points in this order, and for each point k, compute its *expected* distance to its *closest neighbor* among the points $\{1, 2, \ldots, k-1\}$.
- Output the sum of these expected nearest neighbor distances as an approximation to $\mathbb{E}(MST)$.

If all the points of S were *deterministic*, this algorithm would compute an $O(\log n)$ factor approximation of the MST. For a proof of approximation, please see the appendix (this is in fact a known greedy strategy for approximating the Steiner tree of an online sequence of points in a metric space [2, 16]).

When the points of S are stochastic, we use this algorithm to compute the expected length of the edge that is added when the kth point is considered by the algorithm. Let $\mathbb{E}[e_k]$ denote this expected length. Since the points are independent, we can compute this length as follows. Sort the points that precede k in the given order by increasing distance from k and assume, without loss of generality, that the sorted sequence is $\{1, 2, \ldots, k-1\}$. Then, the expected length is given as:

$$\mathbb{E}[e_k] = \sum_{i=1}^{k-1} (\prod_{j=1}^{i-1} (1 - p_j))\ell(k,i),$$

where $\ell(k,i)$ is the distance between the points k and i. (That is, the point j is selected when all the points closer to k are absent.) The expected length for each k can be computed in $O(n \log n)$ time, dominated by sorting, by spending $O(1)$ time per point. Our approximation of the expected MST length is then $\mathbb{E}[A] = \sum_{i \in S} p_i \mathbb{E}[e_i]$. The entire algorithm runs in $O(n^2 \log n)$ time, and we have the following results.

Theorem 4.1. *Given a stochastic set of n points in a metric space or Euclidean d dimensions, we can estimate the expected cost of its MST (or the TSP or the Steiner tree) within $O(\log n)$ factor in time $O(n^2 \log n)$.*

5 Stochastic Maximum Flow Problem

We prove that in a stochastic graph where each edge fails with an independent probability, it is $\#P$-Hard to approximate the expected weight of the maximum flow from a source node s to a sink t. We will later use this result to prove the hardness of stochastic linear programs. Suppose we are given a graph $G = (V, E)$, where each edge $e_i = (u, v)$ has a capacity constraint $c(u, v)$ and a failure probability p_i. The flow is a mapping $f : E \to R^+$ satisfying the constraints $f(u, v) \leq c(u, v)$ for all the edges, and the flow conservation at all the nodes, except s, t. For more information on flow networks refer to the book [8]. The following lemma shows that the stochastic max-flow is $\#P$-Hard by a reduction from the 2-terminal network reliability problem defined earlier in Section 3 (or see [26, 27]). The proof is in the appendix.

Lemma 5.1. *Given a capacitated graph $G = (V, E, c)$, a source s, a sink t, and a rational failure probability p_i for each edge $e_i \in E$, computing the expected value of the s-t max-flow is $\#P$-Hard.*

6 Stochastic Linear Programming

Linear programming is a powerful tool in optimization and geometry. Models for stochastic linear programming typically involve random variables, either in the constraints or in the objective function, for which only the probability distribution is known [20]. This refers to random coefficients or additive terms drawn from partially or fully known probability distributions. In our stochastic model of the linear program, we assume that each linear constraint is active with an independent but arbitrary rational probability. More formally, we have a linear program with n variables and m constraints:

$$\begin{array}{ll} \text{maximize} & c.x \\ \text{subject to} & a_i x \leq b_i, \quad i = 1 \ldots m \\ & x \geq 0 \end{array}$$

where the ith constraint is *active* with probability p_i. We may wish to determine the probability that such a stochastic linear program is *feasible*, or to compute the *expected value* of a given linear objective function. When estimating the expected objective value, we may either assume that the linear program is always feasible, or assign the objective value zero to any infeasible instance. We also assume that the linear program is bounded, for instance, enclosed in a large enough hypercube. While it is well known that the linear programming problem without the probabilistic constraints can be solved in polynomial-time (see [25]), and while several stochastic variants such as chance-constrained linear programming are known in the literature [9, 11], we are unaware of any results on the computational complexity for our basic stochastic model. We show below that for arbitrary dimensions n, the problem is $\#P$-Hard.

6.1 Hardness of Stochastic Linear Programming

We reduce the stochastic max-flow, defined in the previous section, to the stochastic linear programming problem, which immediately proves the hardness of the latter. Consider the max-flow problem, letting p_{ij} be the failure probability of the edge (i, j). The following is a stochastic LP formulation of the max-flow in a stochastic network.

$$\begin{aligned}
\text{maximize} \quad & \sum_{j:(s,j)\in E} f_{sj} && (1)\\
\text{subject to} \quad & f_{ij} \le c_{ij} \qquad \forall (i,j) \in E && (2)\\
& f_{ij} \le 0 \quad \text{with probability } p_{ij}, \qquad \forall (i,j) \in E && (3)\\
& \sum_{j:(i,j)\in E} f_{ij} = \sum_{j:(j,i)\in E} f_{ji} \quad \forall i \in V, i \neq t && (4)\\
& f_{ij} \ge 0 && (5)
\end{aligned}$$

Constraint (3) is the only stochastic constraint, which forces the capacity of an edge (i, j) to be zero, encoding the failure of the edge (i, j) in the flow network. The expected value of this stochastic LP is the solution to our stochastic max-flow problem, and therefore we have our hardness result.

Theorem 6.1. *Given a linear program with n variables and m constraints, such that each constraint i only needs to be satisfied with probability p_i, it is #P-Hard to compute the expected optimal value of the objective function, where the expectation is over all the subsets of valid constraints.*

This readily proves the feasibility problem hard as well: simply set the objective function to 1. (We adopt the conversion that when the LP is infeasible, its objective value is assumed to be zero.)

6.2 Approximating Stochastic LP's and Fixed Dimensional LP

By drawing sample linear programs according to the probability distribution induced by the constraints, we design a fully polynomial randomized (ϵ, δ)-approximation scheme to estimate the probability of the LP feasibility. Let f be a function from some domain D into the positive reals. By an (ϵ, δ)-approximation algorithm for f, we mean a randomized algorithm that for every input $w \in D$, returns an approximation $\tilde{f}(w)$ such that [19, 7]

$$\mathbf{Pr}\left(\frac{|\hat{f}(w) - f(w)|}{f(w)} \;>\; \epsilon\right) \;<\; \delta.$$

The algorithm is fully polynomial if its running time is polynomial in $1/\epsilon, 1/\delta$ and the length of the encoding of w, for every $\epsilon > 0$, and $0 < \delta \le 1$.

Let S denote the set of all the possible 2^m linear programs, depending on which subset of the m constraints are active, and let $A \subseteq S$ be the set of all the feasible programs. Let L_i denote an arbitrary member of S, and let C_i be the set of constraints that must be satisfied in L_i, and let $\tilde{C}_i$ be the complement set of constraints (the ones that are not active). The constraint probabilities induce a probability distribution on S, namely, the probability for the linear program L_i to arise is:

$$P(L_i) = \prod_{c_i \in C_i} p(c_i). \prod_{c_i \in \tilde{C}} (1 - p(c_i)).$$

We now draw a sample of constraints by choosing the ith constraint with probability p_i. Each set of constraints in the sample is a random subset of constraints, which we call the instance L_j. If we draw N such sample instances, and use the random variable $X_j, 1 \leq j \leq N$, which is set to 1 if L_j is feasible, then, we can estimate the probability p that the stochastic linear program is feasible by

$$r = \frac{1}{N} \sum_{1 \leq i \leq N} X_i.$$

We now use the following Chernoff bound to analyze the approximation error in our estimate.

Lemma 6.2 (*Chernoff-Hoeffding Bound*)**.** *Let $X_1 \ldots X_n$ be i.i.d. 0-1 random variables with expectation $E[X_i] = \mu$, for all i. Let $\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$. Then for $0 < \epsilon < 1$ we have*

$$\mathbf{Pr}[(1-\epsilon)\mu \ \leq \ \bar{X} \ \leq \ (1+\epsilon)\mu] \ \geq \ 1 - 2e^{-\frac{\epsilon^2 n\mu}{3}}.$$

Let p be the probability that the linear program is feasible. Since $E[r] = p$, for $N \geq 3/(\epsilon^2 p) \ln(2/\delta)$ using the above bound we have

$$\mathbf{Pr}\left(p(1-\epsilon) \leq r \leq p(1+\epsilon)\right) \ \geq \ 1 - \delta.$$

We summarize this result as follows.

Theorem 6.3. *Given a linear program with m constraints, such that each constraint i only needs to be satisfied with probability p_i, we can obtain an (ϵ, δ)-approximation algorithm that runs in time polynomial in $1/\epsilon$, $\ln(1/\delta)$, and $1/p$ for the probability p that the linear program is feasible.*

When the dimension d (the number of variables) is fixed, one can enumerate all possible nodes of the stochastic polytope. Since m hyperplanes in d-space, form an arrangement of size $O(m^d)$, we can evaluate the probability of each of those nodes being the optimum. We omit easy details from this abstract and summarize the theorems.

Theorem 6.4. *Given a stochastic linear program with m constraints in d variables, we can compute the probability that the linear program is feasible in $O(m^{d+1})$ time.*

Theorem 6.5. *Given a stochastic linear program with m constraints in d variables, we can compute the expected value of the optimum objective function in $O(m^{d+1})$ time.*

Proof. Follows from the linearity of expectation: $E[f] = \sum_{v_i \in V} p(v_i) f(v_i)$, where $p(v_i)$ is the probability that vertex v_i in the arrangement formed by the hyperplanes is optimum. □

7 Closing Remarks

In this paper, we introduced a simple model of stochasticity for some classical optimization problems. Our results show that the introduction of probabilities in input can change the complexity landscape in surprising ways. In the model where each node (edge) of a graph is present probabilistically, we showed that computing the expected cost of the MST (max-flow) is $\#P$-hard.

In the case of metric graphs, we showed that it is possible to approximate the expected MST cost in polynomial time to a logarithmic factor.

We studied a simple model of stochasticity for linear programs in which each constraint is only enforced with a certain, independent probability, and the goal is to find the expected optimum of the objective function or the probability that the program is feasible.

We believe that this simple model has many applications in real world, and can be applied to many other classical combinatorial and geometric problems.

References

[1] D. Aldous and J. M. Steele. Asymptotics for euclidean minimal spanning trees on random points. *Probability Theory and Related Fields*, 92(2):247–258, 1992.

[2] N. Alon and Y. Azar. On-line steiner trees in the euclidean plane. In *SCG '92: Proc. Eighth Annual Symposium on Computational Geometry*, pages 337–343, 1992.

[3] J. Beardwood, J. H. Halton, and J. M. Hammersley. The shortest path through many points. *Proc. Cambridge Philos. Soc.*, 55:299–327, 1959.

[4] D. Bertsimas. *Probabilistic Combinatorial Optimization Problems*. PhD thesis, Operation Research Center, MIT, Cambridge, MASS, 1988.

[5] B. Bollobás. *Current Trends in Stochastic Geometry: Likelihood and Computation*. Cambridge University Press, 2001.

[6] B. Bollobás. *Random Graphs*. Cambridge University Press, 2001.

[7] A. Z. Broder. How hard is it to marry at random? (on the approximation of the permanent). In *STOC '86: Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 50–58, New York, NY, USA, 1986. ACM.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2001.

[9] G. B. Dantzig. Linear programming under uncertainty. *Manage. Sci.*, 50:1764–1769, 2004.

[10] K. Dhamdhere, R. Ravi, and M. Singh. On two-stage stochastic minimum spanning trees. In *IPCO*, volume 3509, pages 321–334, 2005.

[11] M. Dyer and L. Stougie. Computational complexity of stochastic programming problems. *Math. Program.*, 106(3):423–432, 2006.

[12] P. Erdös and A. Rényi. On random graphs. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[13] A. D. Flaxman, A. Frieze, and M. Krivelevich. On the random 2-stage minimum spanning tree. In *SODA '05: Proc. 16th Annual ACM-SIAM symposium on Discrete algorithms*, pages 919–926, 2005.

[14] A. Frieze. On the value of a minimum spanning tree problem. *Discr. App. Math.*, 10:47–56, 1985.

[15] A. Frieze and J. Yukich. Probabilistic analysis of the traveling salesman problem, in *The Traveling Salesman Problem and its Variations*, g. gutin and a.p. punnen (eds.), 2002.

[16] M. Imase and B. M. Waxman. Dynamic steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.

[17] N. Immorlica, D. Karger, M. Minkoff, and V. S. Mirrokni. On the costs and benefits of procrastination: approximation algorithms for stochastic combinatorial optimization problems. In *SODA '04: Proc. 15th Annual ACM-SIAM symposium on Discrete algorithms*, pages 691–700, 2004.

[18] P. Jaillet. A priori solution of a traveling salesman problem in which a random subset of the customers are visited. *Math. Oper. Res.*, 6(6), 1988.

[19] M. Jerrum. Random generation of combinatorial structures from a uniform distribution (extended abstract). In *Proceedings of the 12th Colloquium on Automata, Languages and Programming*, pages 290–299, London, UK, 1985. Springer-Verlag.

[20] P. Kall and J. Mayer. *Stochastic Linear Programming, Models, Thoery, and Applications.* Springer, 2005.

[21] R. M. Karp. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. *Mathematics of Operations Research*, 2(3):209–224, 1977.

[22] I. Katriel, C. Kenyon-Mathieu, and E. Upfal. Commitment under uncertainty: Two-stage stochastic matching problems. *Theoretical Computer Science*, 408(2-3):213 – 223, 2008.

[23] B. Morgen. *An Introduction to Bayesian Statistical Decision Processes.* Prentice-Hall Inc., 1968.

[24] A. G. M. Pal, R. Ravi, and A. Sinha. Boosted sampling: Approximation algorithms for stochastic optimization. In *Proc. 36th Annual ACM Symposium on Theory of Computing*, pages 417–426, 2003.

[25] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, 1982.

[26] J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.*, 12(4):777–788, 1983.

[27] A. Rosenthal. Computing the reliability of complex networks. *SIAM J. Appl. Math*, 32(2):384–393, 1977.

[28] J. Steele. On frieze's $\zeta(3)$ limit for lengths of minimal spanning trees. *Ann. Prob.*, 9:365–376, 1987.

[29] C. Swamy and D. B. Shmoys. Approximation algorithms for 2-stage stochastic optimization problems. *SIGACT News*, 37(1):33–46, 2006.

[30] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

A Proofs

Proof of Lemma 2.2:

Proof. Consider a subset S of $V = V'$. We consider two cases, depending on whether s and t are connected or not in the subgraph induced by S. If the subgraph induced by S leaves s and t connected in G, then s and t are also connected in G' using edges of cost 0 when S is the active set. Therefore, the edge (s, t), which has cost $L > 0$, does not belong to the MST of S. On the other hand, if s and t are disconnected in the subgraph of G induced by S, then every path from s to t includes an edge of cost L or more when S is the active set in G'. Since (s, t) is the edge with the smallest positive cost in G', it necessarily belongs to the MST. Thus, (s, t) belongs to the MST of G' for an active subset S if and only if s and t are not connected in the subgraph of G induced by S. This completes the proof. □

Proof of Lemma 2.4:

Proof. If the subset of S_v nodes forms a vertex cover, then any vertex in S_e is connected to at least one node in S_v using an edge of weight 0. Since all the nodes in S_v are connected to each other via edges if weight 0 as well, no edges of weight larger than 0 are needed in the MST. On the other hand, if the S_v nodes do not form a vertex cover, then there exists at least one node in S_e which is not connected to any surviving node in S_v with an edge of weight 0, and therefore the minimum spanning tree would need at least one edge of weight L, i.e., $MST(H) \geq L$ □

Proof of Lemma 3.2:

Proof. Consider an arbitrary edge $e_k = (v_i, v_j)$ in E. If e_k survives in H, then the corresponding representative point r_k is present in $S(H)$, and the virtual node v_i is connected to the virtual node v_j using a short path of length 1 edges. If, however, e_k fails, then r_k is absent, and the two virtual nodes v_i and v_j can be connected using short edges in $S(H)$ if and only if there is a path connecting v_i and v_j in H—this follows because the virtual nodes in $S(H)$ are at least distance 2 apart, and a path in G maps to a path of short edges in $S(H)$.

Now, let us consider the nodes s and t. If they are connected in G, we have a path of short edges in $S(H)$ connecting their virtual nodes. If s and t are disconnected, then the MST of $S(H)$ must use the edge $\hat{st}$ because this is the shortest edge in the point set $S(H)$ that is longer than 1. This completes the proof. □

Proof of Lemma 3.3:

Proof. By moving $\hat{s}$ and $\hat{t}$, we only affect the lengths of the edges incident to these two points. Since each virtual node is at distance two from another virtual node, the MST never uses any edge longer than two. The edges induced by a subset $S(H)$ can be classified into 3 groups, in ascending order of length: short edges of length 1, the edge $\hat{st}$, and edges of length at least 2. Since the length of $\hat{st}$ in both cases remains strictly between 1 and 2, the relative order of these edges is unchanged, and therefore by the minimum spanning tree property (that the shortest edge across any cut is included in the MST and the longest edge in any cycle is excluded), we conclude that both MST_1 and MST_2 contain exactly the same set of edges, with only the length of $\hat{s}\hat{t}$ changing between them. Thus, the probability $p(a, b)$ remains the same for all pairs a, b for inclusion in MST_1 and MST_2.

□

Proof of the Greedy Approximation:

Proof. Suppose the MST of S has length ℓ, and consider the points S_k for which the greedy algorithm pays more than $2\ell/k$; that is, the points whose nearest neighbor (among the previously arrived points) is at distance $2\ell/k$ or more. We claim that this occurs at most k times—otherwise since all these $|S_k|$ points have pairwise distances at least $2\ell/k$, the traveling salesman tour through them has length at least $|S_k|2\ell/k$. On the other hand, since the MST is a 1/2-approximation of the TSP, we also have that $|S_k|2\ell/k \leq 2\ell$, which proves $|S_k| < k$. With this fact, it follows that the kth largest edge added by the greedy algorithm is at most $2\ell/k$, and so the total length of the tree is at most $\sum_{k=1}^{|S|} 2\ell/k = O(\ell \log |S|)$. □

Proof of Lemma 5.1:

Proof. Given an instance $G = (V, E)$ of the 2-NRP, start node s, destination t, and edge failure probabilities p_i, our stochastic flow graph is the same as G, with unit edge capacities. By adding two artificial nodes s' and t', and edges (s', s) and (t, t') with capacities 1 and failure probabilities 0, we get a flow graph in which the max-flow is 1 when there is a path from s to t, and 0 otherwise. Thus, the expected value of the max-flow from s' to t' is precisely the probability that s and t are connected in G. This completes the proof. □