# Hedging Uncertainty: Approximation Algorithms for Stochastic Optimization Problems

R. Ravi* A. Sinha†

July 10, 2003

**Abstract**

*Stochastic programming* refers to the general class of optimization problems when the inputs have uncertainty, modeled by probability distributions of the input variables. *Two-stage stochastic optimization with recourse* is a widely used framework for stochastic optimization (See, e.g., the recent text by Birge and Louveaux [1]). In this model, a set of input parameters (e.g. future demand) are uncertain, but their distribution is given as a set of discrete *scenarios* (sets of values), each with a fixed probability. The decision variables can be set in two stages - once before the revelation of the scenario, and then again in recourse after the revelation. The second stage decisions are typically costlier to make compared to their first stage counterparts, since they typically involve rapid reaction to the revealed scenario. The objective is to minimize the sum of first stage decision costs and the expected cost of the recourse decisions (taken over the distribution of the scenarios).

We formulate two-stage stochastic models of widely studied problems in discrete optimization, such as the uncapacitated facility location problem. Here, we need to build some facilities in the first stage, after which the demand is revealed. At this point, we are allowed to extend our existing solution with second stage (recourse) facilities. These could be much more expensive, thus providing a motivation for hedging for the uncertainty of the demand by purchasing some first stage facilities. The goal is to minimize the sum of first-stage and expected second-stage facility opening costs and the expected connection cost of the revealed demands to (first or second stage) facilties which

*GSIA, Carnegie Mellon University. ravi@cmu.edu. Supported in part by NSF grant CCR-0105548 and ITR grant CCR-0122581 (the ALADDIN project).

†GSIA, Carnegie Mellon University. asinha@andrew.cmu.edu. Supported in part by a CBI graduate fellowship, NSF grant CCR-0105548 and ITR grant CCR-0122581 (the ALADDIN project).

serve them. We provide a constant-factor approximation algorithm for stochastic facility location with metric distances.

We also show how the stochastic extension of classical problems, such as shortest paths, bin packing, vertex cover and set cover, can be well-approximated in polynomial time, sometimes by surprising connections to problems which are known to have approximation algorithms (E.g., stochastic versions of the classical shortest-path problem are closely related to the Connected Facility Location and group Steiner Tree problems). We believe that the two-stage stochastic optimization model deserves more study motivated by the inherent uncertainty of data in many applications of these problems.

# 1 Introduction

Consider a facility location problem faced by a company which is entering a new country, and needs to install manufacturing facilities to serve the demand of its products. Traditionally, this has been modeled as the uncapacitated facility location (UFL) problem, which aims to minimize the sum of facility opening costs and the sum of distances from clients to their nearest open facility. Now suppose the company doesn't know the demand pattern for certain; instead, it has forecasts of three different scenarios (eg, good, average and bad economy) which might occur, with each scenario inducing a different demand for the product at each city. The two-stage stochastic variant of UFL attempts to capture this uncertainty by allowing the company to open some facilities before the demand has been observed, and augment the solution by opening some more facilities in a second stage after the demand has been observed. However, if the company decides to defer opening facilities until after the demand has been observed, the facilities might become much costlier (or even unavailable).

The company wants to purchase some facilities today, because deferring all facility installations to the future might increase its facility costs significantly. However, it could be overkill to purchase facilities only in the first stage, because many of them may be underutilized. A reasonable model to capture this trade-off is to attempt to minimize the sum of the cost of the facilities opened in the first stage, and the expected cost of the second-stage facilities plus the service costs of the revealed demand clients. In this example, the expectation is taken over the discrete uniform probability distribution of the demand scenarios that is specified in advance. The resulting optimal solution would then be a set of first stage facilities of low cost which hedges against the future uncertainty, and can be augmented in the second stage after the demands have materialized to minimize expected costs overall. This is the standard approach in the stochastic programming community [1, 11]. We present a constant-factor approximation algorithm for the stochastic facility location problem, and stochastic variants of other familiar optimization problems.

## 1.1 Our contributions

- We initiate the study of approximation algorithms for stochastic optimization, by studying some classical combinatorial optimization problems with stochastic inputs under the simple two-stage model. We present this model to the community as a realistic and novel way to

deal with the inherent uncertainties involved in several real-world optimization problems.

- We provide a constant-factor approximation algorithm for stochastic facility location.
- We show that the shortest paths problem, which is extremely simple with deterministic inputs, is NP-hard in the stochastic model. We provide an $O(1)$ approximation for it by reducing it to a network-design problem (equivalent to connected facility location). We then consider the case when the metric is also allowed to change arbitrarily across scenarios (and not just scale proportionately) and show that a special case of this problem models the group Steiner problem; we extend existing techniques to derive a polylogarithmic approximation for this case.
- We provide a 2-approximation algorithm for stochastic vertex cover, showing that the approximability of the stochastic version of the problem matches the deterministic version.
- We also show how the model can be applied to some other classical problems, such as bin packing and set cover.

We have chosen these specific problems to illustrate the rich variety of outcomes in the application of approximation algorithm methods to two-stage stochastic models of classical problems.

## 1.2 Previous Work

**Existing work on approximation algorithms for stochastic optimization** Stochastic inputs have been studied to a limited extent in approximation algorithms. Skutella and Uetz [26] and Mohring, Schulz and Uetz [19] studied various scheduling problems when the job sizes are stochastic. Karger and Minkoff [13] studied the problem of constructing a single Steiner tree when each node has an associated probability of requiring to be connected to the root. The cost is the expected cost of the subtree that is used to connect the randomly materializing terminals via this tree. None of these models incorporates two stages or recourse, and hence they do not fit in our framework.

Some models work under the assumption that each client reveals its demand with a fixed probability independent of other clients. This is not

always realistic - for example, in the facility location problem, there are several macro-economic factors influencing demand, such as the state of the economy, competition, and technology. This is why in practice the scenario model followed in this paper is often used, since a scenario can take into account the correlations between these macro-level factors in establishing the demand pattern. Furthermore, in many problems, the stochastic version with independent probabilities of clients revealing demands can be easily solved (E.g., facility location). We mention some of these cases in this paper at various points when we examine our specific stochastic optimization problems.

**Relation to approximation algorithms** We have chosen some classical problems to illustrate the applicability of stochastic optimization. Uncapacitated facility location under metric costs has fueled much work in approximation algorithms, due to the applicability of almost all approximation techniques to it. Our work is also very naturally motivated by the inherent uncertainty in demand predictions in such facility location planning problems. Shmoys, Tardos and Aardal [25] gave the first constant-factor approximation algorithm for it, and our work builds on their algorithm. A string of papers on this problem culminated in the current best 1.52 approximation due to Mahdian, Ye and Zhang [18]. We present an 8-approximation algorithm for Stochastic Facility Location[1].

The shortest paths problem was one of the first for which fast, efficient exact algorithms were developed, such as Dijkstra's algorithm [4]. Yet, our result shows that adding uncertainty makes it a non-trivial NP-hard problem, showing an interesting behavior of the model we adopted.

The set cover problem was among the first problems shown to be NP-hard, and an $O(\log n)$ approximation was first provided by Johnson [10], which is essentially the best possible. Bin-packing is another well-studied NP-complete problem, with several $O(1)$ approximations known (see [3] for a survey), and so is vertex cover [20]. We show how existing approximation techniques can be adapted naturally to derive results typically with no or very little worsening of performance ratios for the stochastic versions of these problems.

**Dealing with incomplete global knowledge** Classical algorithms are designed to operate with complete knowledge of all input parameters. However, the algorithms community has been well aware of the deficiencies of such a model when applied to the real world. Online algorithms (which

[1] We made no attempts to optimize this factor and believe it can be improved.

do competitive analysis when the input is revealed over time) [2], non-clairvoyant algorithms (where the data is revealed only after the decisions have been made) [12], and game-theoretic algorithms (which offer incentives to induce participants to truthfully reveal their private information) [21] are some ways of dealing with incomplete information which have been recently studied. The model we study offers another approach, often used in practice.

# 2 Stochastic facility location

## 2.1 Definition

As in the classical uncapacitated facility location problem, we are given a set of facilities $F$ and a set of clients $D$, with a metric $c_{ij}$ specifying the distances between every client and every facility. However, the demand of each client is not known at the first stage. In scenario $k$, client $j$ has demand $d_j^k$, which may be zero. Facility $i$ has a first-stage opening cost of $f_i^0$, and recourse costs of $f_i^k$ in scenario $k$. These may be infinity, reflecting the unavailability of the facilities in various scenarios. We abbreviate this problem as SFL.

$$\min \sum_{i \in F} f_i y_i^0 + \sum_{k=1}^{m} p_k \Big( \sum_{i \in F} f_i^k y_i^k + \sum_{i \in F, j \in D} d_j^k c_{ij} x_{ij}^k \Big) \qquad (IP_{SFL})$$

$$\text{s.t.} \sum_{i \in F} x_{ij}^k \geq d_j^k \quad \forall j \in D, \forall k$$

$$x_{ij}^k \leq y_i^0 + y_i^k \quad \forall i \in F, \forall j \in D, \forall k$$

$$x, y \text{ non-negative integers}$$

The problem is best explained by the integer program formulation $IP_{SFL}$ above. While our algorithms extend to arbitrary demands at every client, for simplicity we only study the case when all $d_j^k$'s are either 0 or 1. Variable $x_{ij}^k$ is 1 if and only if client $j$ is served by facility $i$ in scenario $k$. If $x_{ij}^k = 1$, then facility $i$ must either be opened at the first stage ($y_i^0 = 1$) or in recourse in scenario $k$ ($y_i^k = 1$) (or both).

## 2.2 Non-triviality of the problem

First notice that if the second stage facility costs were identical to those in the first stage for all scenarios, then we can "de-couple" the stochastic components of the problem and solve for each scenario independently. On the other hand, if there was no second stage and all facilities had to be opened in the first stage, then SFL reduces to an instance of the usual UFL, where the probability multipliers in the expected service costs can be incorporated into the demand terms (thinking of the demands as being scaled based on the probability of occurrence). This also extends to allowing arbitrary demand distributions at the vertices, if they are independent. In this case, existing approximations for UFL apply directly. The added difficulty, and indeed the interesting aspect of the model, arises from varying (and typically increased) second-stage facility costs under different scenarios.

In the other direction, SFL can be viewed as a special case of the multi-commodity facility location problem (MCFL), where we treat each scenario as a distinct commodity and the cost of a facility depends on the commodities it serves. However, the best-known approximation ratio for MCFL is $\emptyset(\log m)$ [23] (where $m$ is the number of scenarios), so we need different techniques for better approximations of SFL.

The main difficulty stems from the fact that we cannot treat each scenario by itself, since the different scenarios interact in utilizing first-stage facilities. A simple heuristic is to compare the solution obtained if all the demand is satisfied in the first stage with the solution when no first stage facilities are opened. While this heuristic works well in certain instances (particularly, maximization problems such as maximum weight matchings [14]), it can easily be shown to perform badly in our case, due to the interaction across scenarios.

## 2.3 Algorithm

Our approximation algorithm proceeds along the lines of the LP-rounding algorithm due to Shmoys, Tardos and Aardal [25], with some crucial differences. We begin by solving the linear relaxation of $IP_{SFL}$. Let $(x, y)$ denote an optimal LP solution. The first step in rounding this fractional solution is using the filtering technique of Lin and Vitter [16]. We fix a constant $0 < \alpha < 1$. For every client-scenario pair $(j, k)$, we define its optimal fractional service cost to be $c^*_{jk} = \sum_i c_{ij} x^k_{ij}$. Order the facilities which serve the pair $(j, k)$ according to non-decreasing distance from $j$. The $\alpha$ point $g_{j,k}(\alpha)$ for the client-scenario pair $(j, k)$ is the smallest distance $c^\alpha_{jk}$ such

that $\sum_{i:c_{ij} \leq c^{\alpha}_{jk}} x^k_{ij} \geq \alpha$.

Treating each client-scenario pair as a distinct client, we appeal to the following theorem due to Shmoys, Tardos and Aardal. It can be proved along the lines of the original proof in [25], using the observation that we can treat each scenario independently.

**Theorem 1** *Given a feasible fractional solution $(x, y)$, we can find a fractional solution $(\overline{x}, \overline{y})$ in polynomial time such that (i) $\overline{x}^k_{ij} > 0 \Rightarrow c_{ij} \leq c^{\alpha}_{jk}$ for all $i \in F, \quad j \in D, \quad k = 1, 2, \ldots, m$; (ii) $c^{\alpha}_{jk} \leq \frac{1}{1-\alpha} c^*_{jk}$; (iii) $\overline{y}^k_i \leq \min\{1, \frac{y^k_i}{\alpha}\}$ for all $i \in F, \quad k = 0, 1, \ldots, m$.*

The algorithm in [25] proceeds to iteratively round $x^k_{ij}$ variables for which $c^{\alpha}_{jk}$ is smallest. However, this does not work in our case, because the rounding algorithm might close facilities which are needed for other scenarios $k' \neq k$. Hence we need a rounding algorithm which carefully treats the distinction between stage 1 facility variables $y^0$, and recourse facility variables $y^k$.

We proceed as in earlier algorithms by obtaining an optimal LP solution; In the next step, we progressively choose clients *across all scenarios* with minimum fractional service cost, and neglect to serve other clients conflicting (overlapping in facility utilization) with it by assigning them to be served by this client's serving facility. However, this will not work if the serving facility is not open in this neglected client's scenario. Hence, the main difference is that if a stage 1 facility is opened to serve a client, all clients that conflict with it can be served, while if a stage 2 facility variable is rounded up to serve this client, only those clients in the same scenario that conflict with this client are neglected and assigned to this client. This strategy suffices to pay for all opened facilities by the "disjointness" of the different scenarios' contributions in the objective function, while the rule of considering clients in increasing order of fractional service cost allows us to bound the service cost. Our rounding algorithm is described in detail below. Let $0 < \beta < 1$ be another fixed constant.

1. Initialize $\hat{F}^k = \emptyset$ to be the set of facilities opened in scenario $k$ for $k = 0, 1, \ldots, m$. Mark all client-scenario pairs as "unserved".

2. Let $(j, k)$ be an unserved client-scenario pair with smallest $c^{\alpha}_{jk}$. Consider the following cases, in each case marking $(j, k)$ as "served" and proceeding to the next client-scenario pair. Let $S^0$ be the set of facilities $i$ such that $\overline{x}^k_{ij} > 0 \wedge \overline{y}^0_i > 0$, and $S^k$ be the set of facilities $i$ such that $\overline{x}^k_{ij} > 0 \wedge \overline{y}^k_i > 0$.

(a) If $\sum_{i \in S^0} \overline{y}_i^0 \geq \beta$, let $i$ be the facility such that $f_i^0$ is smallest among all facilities in $S^0$. Move facility $i$ to the set $\hat{F}^0$, and set $\hat{y}_i^0 = 1$. For all other facilities $i' \in S^0 \cup S^k$, set $\hat{y}_{i'}^0 = \hat{y}_{i'}^k = 0$. For client-scenario pairs $(j', k')$ such that there exists a facility $i' \in S^0 \cup S^k$ with $c_{i'j'} \leq c_{j'k'}^{\alpha}$, set $\hat{x}_{ij'}^{k'} = 1$ and mark them as "served".

(b) If $\sum_{i:i \in S^0} \overline{y}_i^0 < \beta$, then we must have $\sum_{i:c_{ij} \leq c_{jk}^{\alpha}} \overline{y}_i^k \geq 1 - \beta$. In this case, let $i$ be the facility in $S^k$ with smallest $f_i^k$. Move facility $i$ to the set $\hat{F}^k$ and set $\hat{y}_i^k = 1$. For all other facilities $i' \in S^k$, set $\hat{y}_{i'}^k = 0$. For clients $j'$ such that there exists a facility $i' \in S^k$ with $c_{i'j'} \leq c_{j'k}^{\alpha}$, set $\hat{x}_{ij'}^k = 1$ and mark them as "served".

3. Facilities in $\hat{F}^0$ are the facilities to be opened in stage 1, and facilities in $\hat{F}^k$ are the facilities to be opened in recourse if scenario $k$ materializes. Clients are served according to the zero-one variables $\hat{x}_{ij}^k$.

**Lemma 1** *The rounding algorithm above produces an integer solution $(\hat{x}, \hat{y})$ which is feasible for $IP_{SFL}$ such that*

*(i) For every client-scenario pair $(j, k)$, we have $\hat{x}_{ij}^k = 1 \Rightarrow c_{ij} \leq 3c_{jk}^{\alpha}$.*

*(ii) $\sum_{i \in F} f_i^0 \hat{y}_i^0 \leq \frac{1}{\beta} \sum_{i \in F} f_i^0 \overline{y}_i^0$.*

*(iii) $\sum_{i \in F} f_i^k \hat{y}_i^k \leq \frac{1}{1-\beta} \sum_{i \in F} f_i^k \overline{y}_i^k$ for all $k = 1, 2, \ldots, m$.*

Proof Sketch: The proof is along the lines of a similar theorem proved in [25]. Hence it is only sketched here, with the details deferred to the complete version of this manuscript.

When a client is assigned to a facility (ie, $\hat{x}_{ij}^k$ is set to 1), we either assign it to a facility within distance $c_{jk}^{\alpha}$, or it is assigned when some other client $j'$ with $c_{j'k}^{\alpha} \leq c_{jk}^{\alpha}$ was being considered. In either case, a simple application of triangle inequality yields $c_{ij} \leq 3c_{jk}^{\alpha}$.

When a facility $i$ is chosen for opening in the first stage (ie, $\hat{y}_i^0$ is set to 1), case 2(a) must have occurred. In that case, we have a sufficiently large fraction ($\beta$) of facilities which have $\overline{y}_i^0 > 0$ which we are shutting, and we can charge the cost of opening $i$ to the fractional solution. A similar argument holds for the case when a facility is opened in recourse in scenario $k$.

The solution produced is also feasible, because we start with a feasible solution $(\overline{x}, \overline{y})$, and in each step, we maintain feasibility by ensuring that a client-scenario pair is marked "served" only when its $x_{ij}^k$ variable is set to 1 (ie, it is assigned to a facility) for some facility $i$. □

**Theorem 2** *There is a polynomial time approximation algorithm with performance ratio* 8 *for SFL.*

Proof: Setting $\alpha = \frac{1}{4}$ and $\beta = \frac{1}{2}$, along with Theorem 1 and Lemma 1, yields the performance guarantee. The running time of the algorithm is polynomial in $|D|, |F|, m$. □

**Extensions** The algorithm easily extends to allowing demands at client-scenario pairs which are positive real numbers instead of just 0 or 1. We may also allow the costs to transport one unit of demand per unit length in different scenarios to be different, motivated by, e.g., different scenarios having different prices of gas. In other words, each scenario has a multiplier $\gamma_k$ such that the distance between $i$ and $j$ in scenario $k$ is $\gamma_k c_{ij}$. Essentially, this can be incorporated into the demand variables $d_j^k$, and the rest of the algorithm proceeds as before to give identical results.

# 3 Shortest paths

**Motivation** Consider a supplier who wishes to ship a single unit of a good to a single destination $t$ from a single source $s$, in a graph where the shipping cost is just the cost of the edge. The solution to this problem is to compute a shortest path from $s$ to $t$, and this can be easily done in polynomial time, for example by using Dijkstra's algorithm [4].

Now consider the following stochastic extension. The supplier does not know in advance what the destination is going to be. In particular, any of $m$ scenarios could materialize, with the destination being $t^k$ in scenario $k$. The supplier could enter into contracts in stage 1 to ship the good along edge $e$ at cost $c_e$, but this might be disadvantageous if the destination turns out to be in the opposite direction. However, if the supplier decides to wait for the scenarios to materialize, then the cost of edge $e$ in scenario $k$ changes to $f_k c_e$, which could be disadvantageous if $f_k$ is large. Hence the supplier might wish to reserve some edges now at cost $c_e$, and augment the network in scenario $k$ if necessary.

**Problem definition** We are given a graph $G = (V, E)$, with metric edge costs $c_e$ and a single source $s \in V$. We also have a set of $m$ scenarios, with scenario $k$ specified by a destination vertex $t_k \in V$, a cost scale factor $f_k$, and a probability $p_k$. A feasible solution is specified by a set of edges $E' \subset E$. The first-stage cost of this solution is $\sum_{e \in E'} c_e$, and in scenario $k$, a second stage solution is a path $P_k$ from $s$ to $t_k$; for the second stage

costs, we assume the edges in $P_k$ bought in the first stage, namely in $E'$, have cost zero, while the remaining edges are increased in cost by factor $f_k$, giving second-stage cost $f_k \sum_{e \in P_k \setminus E'} c_e$. The objective is to compute $E'$ which minimizes the sum of first stage edge costs and expected second stage edge costs. We abbreviate this problem as SSP (stochastic shortest paths).

While it is not obvious that $E'$ even induces a connected component, the following lemma can be proved using the facts that an edge is purchased in the first stage only if it is cheaper to do so than wait for the second stage for the set of scenarios which use it, and that we are in the single source-sink model.

**Lemma 2** *The set of edges $E'$ bought in the first stage in an optimal solution to SSP induces a tree containing the source $s$.*

**Interpretation as a network design problem** Armed with the above lemma, SSP can be interpreted as the tree-star network design problem, defined as follows. In tree-star network design, demand nodes have a demand for $d_j$ units of goods to be shipped to a source. A feasible solution is specified by a tree, with the cost of the solution being $M$ times the cost of the tree (for pre-specified $M$) plus the length of the shortest path from each demand node to the tree, weighted by the demand at the node. A constant-factor approximation algorithm for this problem was first provided by Ravi and Salman [22], and it has also been studied subsequently as the connected facility location problem [13, 15], and the asymmetric VPN design problem [7].

**Theorem 3** *There is a polynomial-time constant-factor approximation algorithm for SSP.*

Proof: SSP is equivalent to the tree-star network design problem, via the following transformation. The fixed cost multiplier of the tree $M$ is set to 1. The demand of each node $t_k$ is set to $f_k p_k$. Now purchasing a tree $T$ in stage 1 for SSP is equivalent to building $T$ in the tree-star problem. The expected second stage cost is exactly $\sum_{k=1}^{m} p_k f_k \text{dist}(t_k, T)$, which is the same as incurred in the tree-star problem when the demand at node $t_k$ is $p_k f_k$. □

The equivalence of SSP and tree-star network design also implies the NP-hardness of SSP. Note that SSP is different from the maybecast problem studied by Karger and Minkoff [13], because in their model, each node is a terminal independently with a certain probability, edge costs do not change

across scenarios, and the solution is required to be a single tree spanning all potential terminals.

### 3.1 Stochastic metric

The problem becomes even more interesting (and harder) when the metric itself is allowed to change arbitrarily across scenarios. This might happen, for example, because shipping by sea becomes much cheaper than air transport in one scenario, and vice-versa in another. The problem is defined exactly as in Section 3, except that the cost of edge $e$ in the first stage is $c_e^0$ and in scenario $k$ is $c_e^k$. We call this the stochastic metric shortest paths (SMSP) problem.

In general, the first-stage component of an optimal solution for SMSP need not be a tree. Consider the following example, where there is only one second-stage scenario. The graph is a path with five vertices $s = v_0, \ldots, v_4 = t$, where $s$ and $t$ are the source and the sink respectively. Let $M$ be a large constant. The costs of the four edges $(v_0, v_1), \ldots, (v_3, v_4)$ in the first stage are respectively $1, M, 1, M$, and in the second stage are $M, 1, M, 1$. The optimal solution is clearly to purchase edges $(v_0, v_1)$ and $(v_2, v_3)$ in the first stage, and the others in the second stage; this solution has cost 4. Any solution which requires the first stage to be a tree has cost at least $M$.

**Hardness** Even with the restriction that the first stage set of edges form a tree, SMSP is as hard as the group Steiner tree problem (GST), defined as follows. $G = (V, E)$ is an undirected graph with edge weights $c_e$, and there are $m$ vertex subsets (called groups) $S_k$. The objective is to compute a minimum cost tree which includes at least one vertex from every group. This problem was studied by Garg, Konjevod and Ravi [6] who also gave an approximation algorithm with performance ratio roughly $O(\log^2 n \log m)$, and recently Halperin and Krauthgamer [8] showed an inapproximability threshold of $\Omega(\log^2 n)$ even when $G$ is a tree. For the rest of this section, we consider the restriction of SMSP where the first stage solution has to be a tree, which we dub *Tree-SMSP*. An $\Omega(\log^2 n)$ hardness for Tree-SMSP follows from the reduction of GST to Tree-SMSP, shown below.

**Theorem 4** *An instance of GST can be modeled as a special case of Tree-SMSP.*

Proof: Suppose we are given an instance of group Steiner tree, specified by $G = (V, E)$, metric edge costs $c$, and groups $S_1, S_2, \ldots, S_m$. We create an instance of SMSP with one scenario for every group. The graph remains

the same, and the first stage edge costs $c^0$ are the same as $c$, the edge costs in the GST instance. In scenario $k$, the metric is as follows. The distance between any two vertices in $S_k$ is zero, and all other distances are infinity. Any vertex in $S_k$ is defined to be the destination $t_k$ for scenario $k$. All scenarios are equally likely.

An optimal solution to this instance of Tree-SMSP must select a first stage tree which includes at least one vertex from each $S_k$, to avoid infinite cost. If the tree includes any vertex in $S_k$, it can be augmented at cost zero to a tree which includes $t_k$ if scenario $k$ materializes. □

**Approximation algorithm** Our approximation algorithm relies on the following IP formulation of Tree-SMSP. Variable $r^k_{uv}$ is 1 if edge $(u, v)$ (in the direction $u \to v$) is part of the path traversed from $t_k$ to $s$ and edge $(u, v)$ is chosen in the recourse solution. Variable $f^k_{uv}$ is 1 if edge $(u, v)$ is chosen in the path from $t_k$ to $s$ and edge $(u, v)$ is part of the first-stage solution. Variable $x_{uv}$ is 1 if edge $(u, v)$ is chosen in the first-stage tree.

$$
\begin{aligned}
(IP_{SMSP}) \quad \min \sum_e c_e x_e \quad & + \sum_{k=1}^{m} p_k \sum_e r^k_e c^k_e \\
\text{s.t.} \quad \sum_v (r^k_{t_k,v} + f^k_{t_k,v}) \quad & \geq 1 \quad \forall k \\
\sum_v (r^k_{uv} + f^k_{uv}) \quad & = \sum_v (r^k_{vu} + f^k_{vu}) \\
& \quad \forall u \in V \setminus \{t_k, s\}, \forall k \\
\sum_v r^k_{uv} \quad & \leq \sum_v r^k_{vu} \\
& \quad \forall u \in V \setminus \{t_k\}, \forall k \\
f^k_e \quad & \leq x_e \\
& \quad \forall e \in E, \quad \forall k \\
f, r, x \quad & \text{non-neg. integers}
\end{aligned}
$$

The third set of inequalities are strengthenings valid only for the tree version of SMSP, insisting that flows along recourse arcs from $t_k$ to $s$ via any node are non-increasing; they are also crucial for obtaining the result below. $IP_{SMSP}$ is polynomial in size, so its linear relaxation $LP_{SMSP}$ can be solved optimally in polynomial time. Let $(f, r, x)$ denote an optimal solution to the linear program $LP_{SMSP}$, and $OPT_{SMSP}$ be its value. The following theorem describes our rounding algorithm, with the full proof deferred to

the full version of this manuscript.

**Theorem 5** *The fractional solution $(f, r, x)$ can be rounded in polynomial time to an integer solution $(\hat{f}, \hat{r}, \hat{x})$ such that its cost is $O(\log^2 n \log m)$ times $OPT_{Tree-SMSP}$.*

Proof Sketch: For each destination $t_k$, let $r^*(k) = \sum_e r_e^k c_e^k$ be the cost incurred by the recourse component of the fractional path for $t_k$. Let $S_k$ be the set of all nodes within distance $2r^*(k)$ of $t_k$ in the metric $c^k$. The idea is that we can incur a factor of 2 and pay for a path from $t_k$ to any node in $S_k$ by charging it to $r^*(k)$, and hence we need a first stage tree which reaches at least one node in $S_k$. We construct sets $S_k$ for every scenario $k$, and create an instance of the group Steiner tree problem using the metric $c$.

Using Markov's inequality, it can be shown that if $s \notin S_k$, then we have $\sum_{e=(u,v):u \in S_k, v \notin S_k} x_e \geq \frac{1}{2}$. Hence $2x$ is a fractional solution to the linear relaxation of the following IP formulation of the group Steiner tree problem: $\min \sum_e c_e x_e$ such that $\sum_{e=(u,v):u \in S, v \notin S} x_e \geq 1 \quad \forall S \; \exists k \; : \quad S_k \subseteq S$. Using the result of [6], we can construct an integer solution $\hat{x}$ at a cost which is no more than $O(\log^2 n \log m \cdot OPT_{SMSP})$ which is a tree and includes at least one vertex of every $S_k$. Since for every scenario $k$ we can augment this tree to include $t_k$ at cost no more than $2r^*(k)$, our approximation ratio follows. □

# 4 Stochastic versions of other classical approximation problems

## 4.1 Bin packing

**Problem definition and algorithm** Stochastic bin packing is motivated by applications where storage capacity has to be reserved in advance of the arrival of the objects, and if the reserved capacity is insufficient, we have to purchase additional capacity at possibly higher costs. Formally, we are given a bin capacity $B$, known in advance. There is a set of $m$ possible scenarios, with scenario $k$ specified by a probability $p_k$ of occurrence, a set $S_k$ of objects (each with size $s_i^k \leq B$), and a bin cost $f_k$. A feasible solution is specified by a number $x$ of bins purchased in stage 1, at unit cost per bin. If scenario $k$ materializes, the objects in $S_k$ need to be packed into bins of capacity $B$, which may necessitate the purchase of an additional number of bins at cost $f_k$ per bin. The objective is to compute $x$ so as to minimize the expected total cost. Let $[x]$ denote the integer nearest to $x$.

Let $\rho$ denote the approximation ratio of some (the best) approximation algorithm for the bin-packing problem. Any locally optimal algorithm (first-fit, for example) achieves $\rho = 2$. An asymptotic PTAS was given by Fernandez de la Vega and Lueker [5], which uses at most $(1 + 2\epsilon)OPT + 1$ bins. The following theorem shows how to extend any bin-packing algorithm to handle stochastic bin-packing.

**Theorem 6** *Order the scenarios so that we have* $\sum_i s_i^1 \geq \sum_i s_i^2 \geq \ldots \geq \sum_i s_i^m$. *Let* $k^*$ *be the largest integer such that* $\sum_{k=1}^{k^*} f_k p_k \geq 1$. *Then* $x = \rho\lceil \sum_i s_i^{k^*} \rceil$ *is an asymptotic* $\rho$*-approximate solution.*

Proof: Consider the fractional relaxation of the problem, when we can pack items fractionally into bins. In that case, $x^* = \lceil \sum_i s_i^{k^*} \rceil$ is the optimal solution, because it is the point where the expected marginal cost of buying an additional bin in recourse goes below 1. The expected total cost if we purchase $x^*$ bins is $x^* + \sum_{k>k^*} p_k f_k(\lceil \sum_i s_i^k \rceil - x^*)$, which is a lower bound on the value of an optimal solution of stochastic bin packing.

Since $\rho\lceil \sum_i s_i^k \rceil$ bins are asymptotically sufficient to pack the objects in $S_k$, we will need to purchase at most $\rho(\lceil \sum_i s_i^k \rceil - x^*)$ additional bins if scenario $k > k^*$ materializes. If scenario $k \leq k^*$ is realized, then $\rho x^*$ bins are sufficient and no additional bins are needed. Hence the expected cost of our solution is $\rho x^* + \sum_{k>k^*} p_k f_k \rho(\lceil \sum_i s_i^k \rceil - x^*)$, which is asymptotically no more than $\rho$ times our lower bound. □

## 4.2 Vertex cover

**Problem definition** We are given an undirected graph $G = (V, E)$. As usual, there are $m$ scenarios. In scenario $k$, vertex $v$ has cost $c_v^k$ and edge $e$ has requirement $d_e^k$ which is 1 if $e$ is required to be covered in scenario $k$, and 0 otherwise. Each vertex also has a first-stage cost, $c_v^0$. The objective is to identify a set of vertices to be selected in the first stage, so that the expected cost of extending this set to a vertex cover for each scenario is minimized.

**Generalization** We generalize the problem defined above, and provide a 2-approximation for the generalized problem. This matches the approximation ratio for classical vertex cover. In the generalization, in addition to the problem specified above, the first stage graph $G_0 = (V, E_0)$ and the second stage graphs $G_k = (V, E_k)$ have possibly different edge-sets, though the vertex set is the same. If vertex $v$ is selected in the first stage and scenario $k$ materializes, then $v$ covers only those edges adjacent to it which are in both

$E_k$ and $E_0$. Edges in $E_k \setminus E_0$ have to be covered by vertices purchased in the second stage.

**Integer program formulation** Our algorithm is a primal-dual algorithm which rounds the natural IP formulation of stochastic vertex cover. Variable $x_v^k$ indicates whether or not vertex $v$ is purchased in scenario $k$ (where $k = 0$ as usual denotes the first stage), and $d_{uv}^k$ indicates the existence of edge $(u, v)$ in scenario $k$. In the program below, $d_{uv}^k = 1$ always, but we retain $d_{uv}^k$ to allow for the extension when some edges may require coverage at both ends.

$$\begin{aligned} \min c^0 x^0 + \sum_{k=1}^{m} p_k c^k x^k \quad & (IP_{SVC}) \\ \text{s.t. } x_u^0 + x_v^0 + x_u^k + x_v^k \quad & \geq d_{uv}^k \\ & \forall uv \in E_k \cap E_0, \forall k \\ x_u^k + x_v^k \quad & \geq d_{uv}^k \\ & \forall uv \in E_k \setminus E_0, \forall k \\ x \quad & \text{non-neg. integers} \end{aligned}$$

**Dual program** The dual of the linear relaxation of $IP_{SVC}$ is shown below. Variable $y_e^k$ packs edge $e$ in $E_k$ if $e \in E_k$, and it packs $e \in E_0$ if $e \in E_k \cap E_0$.

$$\begin{aligned} \max \sum_{k=1}^{m} \sum_{u,v \in V} d_{uv}^k y_{uv}^k & & & & (DP_{SVC}) \\ \text{s.t.} \sum_{e \in E_k : v \in e} y_e^k & \leq & p_k c_v^k & \quad \forall v, \forall k \\ \sum_{k=1}^{m} \sum_{e \in E_0 \cap E_k : v \in e} y_e^k & \leq & c_v^0 & \quad \forall v \\ y & \geq & 0 & \end{aligned}$$

**Algorithm** The algorithm is a greedy dual-ascent type of primal-dual algorithm, with two phases. In Phase I, we raise the dual variables $y_e^k$ uniformly for all edges in $E^k \setminus E_0$, separately for each $k$. All vertices which become tight (have the first dual constraint packed to $p_k c_v^k$) have $x_v^k$ set to 1, and deleted along with adjacent edges. We proceed this way until all edges in $E^k \setminus E^0$ are covered and deleted.

In Phase II, we do a greedy dual-ascent on all *uncovered* edges of $E_k$. These edges are contained in $E_k \cap E_0$. This time around, we use a slightly different rule for purchasing vertices. If a vertex is tight for $x_v^0$ (i.e., second dual constraint packed to $c_v^0$), then we select it in the stage 1 solution, and if it is not tight for $x^0$ but is tight for $x^k$ (packed in the second dual constraint), then we select it in the recourse solution.

**Theorem 7** *The integer program $IP_{SVC}$ can be rounded by a primal-dual algorithm within a factor of* 2 *in polynomial time.*

Proof: We analyze the performance of the algorithm described above. Every edge must become tight in either Phase I or Phase II, by definition of the algorithm. Since at least one of the two end-points are purchased for every tight edge, the algorithm yields a feasible solution. We show the ratio of 2 by arguing that every edge pays at most twice its accumulated dual value to account for the total cost of the chosen primal solution.

We now analyze the cost of this solution. First consider edges in $E^k \cap E_0$. Each such edge pays its dual value at most twice, once at each end, since it only enters the algorithm in Phase II.

We next bound the payment of edges in $E_k \setminus E_0$. Consider one such edge, $e = uv$. Without loss of generality, $e$ went tight at $u$ in Phase I. In that case, vertex $u$ is selected in scenario $k$, paid for only by the dual of $E_k \setminus E_0$, and its stage 1 copy is not packed at all by scenario $k$ since only uncovered edges have their duals grown in Phase II. The same argument holds if $e$ also became tight at $v$.

Now suppose $e$ became tight at $u$ in Phase I, but tightened at (packed) $v$ only in Phase II. In this case, $e$ paid only one copy of its dual in Phase I (for $u$). It needs to pay at most one more copy of its dual, for $v$ (whether in stage 1 or recourse). Again, the dual is charged at most twice. □

### 4.3 Set cover

**Problem definition** There is a universe $U$ of $|U| = n$ elements, and a collection $\mathcal{S}$ of subsets of $U$. Each set $S \in \mathcal{S}$ has a stage 1 cost $c_S^0$ and a cost of $c_S^k$ in scenario $k$, some of which might be infinity reflecting the unavailability of the set in certain scenarios. Each element $u \in U$ has a demand vector $d_u$ with the $k^{th}$ component $d_u^k$ being 1 if it is required to cover $u$ in scenario $k$, and 0 otherwise. A feasible solution is specified by a collection $\mathcal{S}' \subseteq \mathcal{S}$, with stage 1 cost $\sum_{S \in \mathcal{S}'} c_S^0$. If scenario $k$ is realized, then $\mathcal{S}'$ must be extended by purchasing some more sets $\mathcal{S}^k$ to cover all elements

with $d_u^k = 1$. The cost of this recourse solution is $\sum_{S \in \mathcal{S}^k} c_S^k$, and we incur this with probability $p_k$. The objective is a solution which minimizes the sum of first stage and expected second stage costs.

**Reduction to classical set cover** While set cover is known to be inapproximable better than $\Omega(\log n)$, it is easy to show an inapproximability of $\Omega(\log m)$ for stochastic set cover through the simple artifact of associating every element with a distinct scenario. We show below an $O(\log nm) = O(\log n + \log m)$ approximation algorithm by viewing stochastic set cover as an ordinary set cover problem, which matches the inapproximability upto constants. The reduction in Theorem 8 allows us to extend the model to the following generalization, for which the same approximation guarantee holds: In scenario $k$, each set $S_k$ covers only a subset of the elements that the first-stage set $S$ covers.

**Theorem 8** *Any stochastic set cover problem is equivalent to a classical set cover problem with $mn$ elements and $|\mathcal{S}|(m+1)$ sets.*

Proof: Associate an element $u_k$ for every element-scenario pair $(u, k)$ such that $d_u^k = 1$. Create $m + 1$ copies of every set $S \in \mathcal{S}$. Set $S^0$ contains all elements $u_k$ for all $k = 1, 2, \ldots, m$ such that $u \in S$, while set $S^k$ only contains $u_k$ for all $u \in S$. Finally, the cost of $S^0$ is $c_S^0$ and that of $S^k$ is $p_k c_S^k$. □

# 5 Conclusion

We have examined some discrete optimization problems under the two-stage stochastic optimization model with recourse, which is a useful model for practical decision making. We initiate the use of approximation algorithms for problems in this model.

**Open questions** There are several classical combinatorial optimization problems for which algorithms in the two-stage stochastic model are not known. An orthogonal direction of research is to develop polynomial time algorithms for other models of stochastic optimization. For example, the two-stage model can be extended to multiple stages. It would also be interesting to approximate stochastic optimization problems where the uncertainty is modeled by other distributions, particularly continuous ones. Finally, can stronger inapproximability results be proved for stochastic versions of classical problems using the inherent uncertainty that these models can encode?

## References


[1] J. Birge and F. Louveaux. *Introduction to Stochastic Programming*, Springer, 1997.

[2] A. Borodin and R. El-Yaniv. *Online computation and competitive analysis*, Cambridge, 1998.

[3] E. Coffman Jr., M. Garey and D. Johnson. Approximation algorithms for bin-packing: a survey. In D.S. Hochbaum, *Approximation Algorithms for NP-hard Problems*, PWS, Boston, 1997.

[4] E. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik* 1:269-271, 1959.

[5] W. Fernandez de la Vega and G.S. Lueker. Bin packing can be solved within $1 + \epsilon$ in linear time. *Combinatorica* 1:349-355, 1981.

[6] N. Garg, G. Konjevod and R. Ravi. A polylogarithmic approximation algorithm for the group Steiner tree problem. *Journal of Algorithms* 37(1):66-84, 2000.

[7] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi and B. Yener. Provisioning a virtual private network: A network design problem for multicommodity flow. In *Proceedings of the* $33^{rd}$ *Annual ACM Symposium on Theory of Computing*, 389-398, 2001.

[8] E. Halperin and R. Krauthgamer. Polylogarithmic inapproximability. To appear in *Proceedings of the* $35^{th}$ *Annual ACM Symposium on Theory of Computing*, 2003.

[9] H. Heitsch and W. Römisch. Scenario reduction algorithms in stochastic programming. *Computational Optimization and Applications* 24:187-206, 2003.

[10] D. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences* 9:256-278, 1974.

[11] P. Kall and S. Wallace. *Stochastic Programming*, Wiley, Chichester, England, 1994.

[12] B. Kalyanasundaram and K. Pruhs. Speed is as powerful as clairvoyance. *Journal of the ACM* 47(4):617-643, 2000.

[13] D. Karger and M. Minkoff. Building Steiner trees with incomplete global knowledge. In *Proceedings of the* $41^{st}$ *Annual IEEE Symposium on Foundations of Computer Science*, 613-623, 2000.

[14] N. Kong and A. Schaefer. A factor $\frac{1}{2}$ approximation algorithm for a class of two-stage stochastic mixed-integer programs. *Manuscript, submitted to INFORMS Journal of Computing*, 2003.

[15] A. Kumar and C. Swamy. Primal-dual algorithms for connected facility location problems. In *Approximation Algorithms for Combinatorial Optimization*, 256-270, 2002.

[16] J-H. Lin and J. Vitter. $\epsilon$-approximations with minimum packing constraint violation. In *Proceedings of the* $24^{th}$ *Annual ACM Symposium on Theory of Computing*, 771-782, 1992.

[17] F. Louveaux and D. Peeters. A dual-based procedure for stochastic facility location. *Operations Research* 40:564-573, 1992.

[18] M. Mahdian, Y. Ye and J. Zhang. A 1.52 approximation algorithm for the uncapacitated facility location problem. In *Approximation Algorithms for Combinatorial Optimization*, 229-242, 2002.

[19] R. Möhring, A. Schulz and M. Uetz. Approximation in stochastic scheduling: The power of LP-based priority policies. *Journal of the ACM* 46(6):924-942, 1999.

[20] B. Monien and E. Speckenmeyer. Ramsey numbers and an approximation algorithm for the vertex cover problem. *Acta Informatica* 22:115-123, 1985.

[21] N. Nisan and A. Ronen. Algorithmic mechanism design. In *Proceedings of the* $31^{st}$ *Annual ACM Symposium on Theory of Computing*, 129-140, 1999.

[22] R. Ravi and F.S. Salman. Approximation algorithms for the traveling purchaser problem and its variants in network design. In *European Symposium on Algorithms*, 29-40, 1999.

[23] R. Ravi and A. Sinha. Multicommodity facility location. *Manuscript*, submitted to SODA 2004.

[24] R. Schultz, L. Stougie and M.H. van der Vlerk. Two-stage stochastic integer programming: A survey. *Statist. Nederlandica* 50(3):404-416, 1996.

[25] D. Shmoys, E. Tardos and K. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the* $29^{th}$ *ACM Symposium on Theory of Computing*, 265-274, 1997.

[26] M. Skutella and M. Uetz. Scheduling precedence-constrained jobs with stochastic processing times on parallel machines. In *Proceedings of the* $12^{th}$ *Annual ACM-SIAM Symposium on Discrete Algorithms*, 589-590, 2001.

[27] L. Stougie. Design and analysis of algorithms for stochastic integer programming. *CWI Tract* Vol. 37, CWI, Amsterdam, 1987.