Thesis Proposal: Approximation Techniques for Stochastic Optimization Problems

Ravishankar Krishnaswamy
Computer Science Department
Carnegie Mellon University

Abstract

The focus of this thesis is on the design and analysis of algorithms for Stochastic Optimization problems, i.e., combinatorial optimization where there is some form of uncertainty in the input. More specifically, we consider designing approximation algorithms for Stochastic Optimization problems in the fields of Network Design and Scheduling.

The exact problems we study in this thesis are the following: (a) the *Survivable Network Design* problem with a random collection of input terminals, (b) the *Stochastic Knapsack* problem with random sizes/rewards for jobs, (c) the *Multi-Armed Bandits* problem, where the individual Markov Chains make random transitions, and finally (d) the *Stochastic Orienteering* problem, where the tasks at each node have random processing times. We explore different techniques for solving these problems and present algorithms for all the above problems with near-optimal approximation guarantees. We also believe that the techniques are fairly general and have wider applicability than the context in which they are used in this thesis.

1 Introduction

The main focus of this thesis is on the the design and analysis of algorithms for optimization problems of practical relevance, like those arising in network design and scheduling. Since many interesting problems in these domains turn out to be computationally intractable (NP-Hard), we resort to designing efficient approximation algorithms, i.e., those that compute solutions which are provably near-optimal.

While several fundamental problems in this area have seen much progress over the years (often resulting in the development of new insights and solution techniques), the models studied have been stylized and somewhat restrictive. An important modeling assumption that is typically made by most earlier works is that the *exact input* is known in advance by the algorithm. For example, consider the problem of scheduling jobs to machines in order to minimize the makespan (i.e., the maximum load on any machine). There are very good approximation algorithms for this problem, including a PTAS via dynamic programming, and a 2-approximation for a more general model using LP rounding techniques. However, all "classical algorithms", perhaps unreasonably, require knowing the exact sizes of the jobs in advance.

Stochastic Optimization. Over the past few years, we have developed several models to overcome this assumption. The one most relevant to this thesis is the framework of Stochastic Optimization. In this model we assume that the algorithm is given only a *distribution* over possible actual inputs to the optimization problem up front (think of the input as being the processing times of jobs, or which terminals need connectivity, etc.). The actual input is, however, known only at a later stage (e.g., the algorithm knows the actual size of the job only after it has completed the job, etc.). The goal is to design algorithms that handle this form of uncertainty while ensuring good approximation ratio. While there are several different models for how the randomness is revealed over time, and also various metrics for quantifying the performance of an algorithm, we restrict our attention to only those pertaining to the problems we study in this thesis, network design and scheduling.

1.1 Uncertainty in Network Design: Stochastic Optimization with Recourse

The input to Network Design problems is usually a graph and a set of terminals with connectivity requirements among them, and the goal is to build a network of minimum cost (where the costs could be on the edges or vertices of the graph), which satisfies all the connectivity requirements of the terminals. However, very often in such *infrastructure planning*, the algorithm designer is not aware of the exact set of terminals or their connectivity requirements when he/she is building the base network, but rather has only some distributional information available up front. Subsequently, the actual requirements of the terminals are revealed, and the designer needs to take corrective action (typically at inflated prices because these decisions are often made in rapid reaction to the observed scenario), to satisfy the requirements of terminals that the base graph did not meet. This naturally motivates the following model known as *2-Stage Stochastic Optimization with Recourse.*

Definition 1.1 (2-Stage Stochastic Optimization) *The input to the algorithm consists of a probability distribution over possible realizations of the data called* scenarios*, and the goal is to construct a feasible solution for the underlying optimization problem in two stages. In the first stage, the algorithm may take some decisions to construct an anticipatory part of the solution, x, incurring a cost of $c(x)$. Subsequently a scenario A is realized according to the distribution, and in the second stage, the algorithm may augment the initial decisions by taking recourse actions y_A*

(in order to make the combined solution $x \cup y_A$ feasible for the scenario A), incurring a certain (typically inflated) augmentation cost $f(x, y_A)$. The goal is then to choose the initial decisions so as to minimize the expected total cost, $c(x) + E_A[f(x, y_A)]$, where the expectation is taken over all scenarios A according to the given probability distribution.

When applied to network design problems, the solution x typically is a subgraph of G, and the initial cost function $c(x) = \sum_{e \in x} c(e)$ is the total cost of building the edges in the first stage. The set A is the actual collection of terminals (along with connectivity requirements), and the recourse stage involves building/buying more edges (with an inflation factor of σ) so that the subgraph $x \cup y_A$ satisfies all the terminals' requirements. The cost of the recourse stage is then $f(x, y_A) = \sigma \sum_{e \in y_A} c(e)$.

An important aspect which we have not yet explained is how the distribution is given as input to the algorithm. Typical models include listing the possible scenarios along with their probabilities (which is feasible only if the number of them is polynomially bounded), or assuming that there is a black-box oracle from which the algorithm can draw samples. A third model is to assume that each vertex has some individual probability of requiring connectivity, and these are independent across vertices.

Related Work. Two-Stage Stochastic Optimization problems have been widely studied in both the Computer Science and Operations Research literature, starting with the works of Dantzig [Dan04], and Beale [Bea55] for Stochastic Linear Programming. Due to the abundance of literature dealing with the efficient computability 2-Stage Stochastic Linear Programming, we refer the interested reader to the book by Birge and Louveaux [BL97] for a more comprehensive discussion. On the other hand, the design and analysis of algorithms for 2-Stage Stochastic Optimization problems (from the point of view of obtaining integral solutions) is relatively less well-understood. The first result in this regard appears to be that of Dye et al. [DST03] who give a constant-factor approximation algorithm for a resource-allocation problem (in the polynomial scenario model). Subsequently, a series of papers [RS06, IKMM04b, GPRS04, SS06a] appeared on this topic in the Computer Science literature, and showed that one can obtain guarantees for a variety of stochastic combinatorial optimization problems by adapting the techniques developed for the deterministic analogue. In particular Gupta et al. [GPRS04] consider the black-box model and show an explicit connection between the approximability of the stochastic problem and the underlying deterministic combinatorial optimization problem.

1.2 Uncertainty in Scheduling: Adaptive Stochastic Optimization

In scheduling problems, uncertainty is often manifested in the processing times of the jobs. Therefore, in the stochastic versions of these problems, the input contains a distribution over possible sizes each job can assume. However, we assume that the size of each job is independent of the others — this is done in order to both make the problem simpler to analyze, and to keep the input representation compact.

In contrast to the long-term infrastructure planning type problems which the Network Design models deal with, the scheduling framework concerns decision making over a more short-term basis, but with the difference that the randomness is not revealed all at once. In particular, a job scheduler will get to know the actual size of a job *only* when it completes. Note that the solutions of our algorithms could now be significantly more complex when compared to the underlying deterministic scheduling problem: While the solution to a deterministic scheduling problem is just an assignment of jobs over time to each machine, the solution to a stochastic scheduling problem is in fact a *scheduling policy* that decides the next job to schedule at any time, *based on the actual sizes to which previously scheduled jobs instantiated*, which is a random variable. Therefore, even to describe the optimal solution given an input may take exponential space, since it could potentially be a complete decision tree of depth n.

Since all our results all concern with scheduling on a single machine, we now describe adaptive solutions and non-adaptive solutions specific to this special case.

In the most general model we study in this thesis, each job is modeled as a Markov Chain with rewards on the states, and there is a budget B on the total time available to the scheduler. Then an adaptive algorithm to schedule the jobs can do the following actions at each timestep:

(i) schedule one unit of a job (not necessarily the job which was scheduled in the previous timestep, i.e., it can generate preemptive schedules). The job then makes a random transition according to its Markov Chain (from its current state), and the algorithm collects a reward from the new state it made a transition to,

(ii) or the budget runs out, at which point the algorithm cannot process any more jobs.

The objective function is to maximize the total expected reward obtained in the time span of B. The above general model captures a wide variety of stochastic scheduling problems, ranging from simple ones like stochastic knapsack to problems, to scheduling problems (with preemptions or cancellations), and finally those with a lot more complexity such as the multi-armed bandits problem.

1.3 A Note on the Models Studied

Before we move on to describing the results of this thesis, we now make a few observations about this model.

1. All problems we study are *offline* problems. The only online aspect to these problems is that the randomness of the input parameters is only revealed over time.

2. The problems are computationally at least as hard as their deterministic counterparts. This is because we allow arbitrary distributions which easily capture deterministic instances. From a modeling perspective, this is different from the average case analysis, where the randomness assumptions are made to make the problem easier than its deterministic counterpart.

3. On the positive side, these problems are easier than their online versions. Intuitively this is because the performance guarantee of our algorithms are measured as the ratio of the expected cost that our algorithm obtains to the expected cost that an optimal algorithm obtains, where the corresponding expectations are taken over the randomness associated with the input and the random coin tosses of the respective algorithm. In particular, this notion is weaker than the *competitive ratio* of the algorithm, which compares the cost of our algorithm with that of the best minimum cost solution for the instance which was revealed. The difference is that in the former expression, OPT is on a "level playing-field" as our algorithm in that both algorithms only know the distributions up front and the randomness is revealed over time. In the latter expression, OPT actually knows the randomness associated with the instance ahead of time.

2 Results in this Thesis

The following list briefly describes the results which are contained in this thesis, along with the algorithmic techniques we use to solve the respective problems.

(a) *Stochastic Survivable Network Design.* In this Two-Stage Stochastic Network Design problem, we are given a graph with associated edge costs, and associated with each pair of vertices $\{s, t\}$ is a probability p_{st} of this pair actually needing connectivity (to an extent of, say, r_{st} which is also given as input). The goal of the algorithm is to buy edges in the first stage, and then augment it with more edges to satisfy the actual demand (which is sampled from the above-mentioned distribution), so as to minimize the total expected cost of the edges bought. Of course, the edges in the second stage are more expensive by a factor of σ.

Techniques. At a high level, our algorithm *reduces* the problem of building the network to the set cover problem over a suitable set system (for which we know good *online algorithms*). More specifically, we construct a set system where the elements are the cuts separating the *s*-*t* pairs, and sets are the edges crossing this cut. While such a system would have exponentially many sets/elements when constructed naively, we can make it much smaller by substantially simplifying the structure of the graph using ideas from low-stretch tree embeddings. We believe that our set-system can have much wider applicability in solving network design problems. In fact, such a reduction even gives us good online algorithms for the survivable network design problem, which immediately results in good algorithms for the two-stage stochastic version of the problem following a reduction due to [GPRS04].

(b) *Stochastic Knapsack.* In this Stochastic Scheduling problem, we are given a collection of jobs, each equipped with a distribution over $(\mathsf{size}, \mathsf{reward})$ pairs. The goal is to adaptively schedule these jobs so as to maximize the profit of jobs that have been completely scheduled within a knapsack of total budget B. However, the actual size/reward of a job are revealed only when it completes.

Techniques. We give constant-factor non-adaptive approximation algorithms for the correlated stochastic knapsack problem improving upon earlier work which only achieves a constant-factor approximation when the random reward of any job is independent of its size. Our approach is to formulate a *time-indexed* LP for the knapsack problem, and show that (a) an optimal adaptive solution (i.e., the probability distributions of OPT placing the items) can be "embedded" as a feasible fractional solution for the LP, and (b) any fractional LP solution can be "rounded" into an integral solution (which can be interpreted as non-adaptive schedule) with only a constant-factor loss in terms of expected total reward in both steps.

(c) *Multi-Armed Bandits.* This is a generalization of the above-mentioned problem where each job is in fact a Markov Chain (which is fully described in the input), and each time we schedule one unit of this job, it makes a random transition according to the Markov Chain (and fetches a reward of the new state it is in). While this problem has been extensively studied from the point of view of (additive) regret, it has only recently received attention from the (multiplicative) approximation algorithms community.

Techniques. We employ similar techniques as mentioned above, the difference being that our rounding algorithm for the general MAB problem is *adaptive* which is in contrast to that for the stochastic knapsack problem. However, we counterbalance this by showing that there can be instances of the MAB problem with arbitrarily large gaps between adaptive and non-adaptive solutions.

(c) *Stochastic Orienteering.* In this variation, the stochastic items are located at nodes on a metric, and there is a total budget of B on traveling plus processing items. We need to visit and process items in order to maximize the total expected reward. This problem generalizes both the deterministic orienteering problem, as well as the stochastic knapsack problem.

Techniques. Unlike the stochastic knapsack problem, we cannot immediately proceed via an LP-based approach to embed an adaptive optimal solution and subsequently rounding it, because we are not aware of any good LP-relaxation for even the deterministic orienteering problem. To circumvent this, we create a surrogate deterministic problem (with appropriately chosen "average" sizes and rewards for items) such that (a) it has a feasible solution with reward at least OPT, and (b) we can convert any feasible solution into a non-adaptive tour with good expected profit. To achieve this, we use a martingale-based analysis on a carefully chosen sequence of random variables.

In the coming four sections, we explain our results and techniques for each of the above problems in more detail.

3 Stochastic Survivable Network Design

In this section, we consider the edge-connectivity version of the *survivable network design* problem (SNDP). In the basic problem, we are given a graph $G = (V, E)$ with non-negative edge-costs $c(e)$, and edge-connectivity requirements $r_{ij} \in \mathbb{Z}_{\geq 0}$ for every pair of vertices $i, j \in V$. The goal then is to find a subgraph $H = (V, E')$ with minimum cost such that H contains r_{ij} edge-disjoint paths between i and j. The cost of a subgraph H is $\sum_{e \in E'} c(e)$.

The problem is of much interest in the network design community, since it seeks to build graphs which are resilient to edge failures. Since the problem contains the Steiner tree problem as a special case (when $r_{ij} = 1$ for every pair of terminal vertices), the general problem is NP-hard, and therefore it has been widely studied from the viewpoint of approximation algorithms (the interested reader may refer to [GK11] for a survey of results and techniques).

In fact, these connectivity problems have served as a test-bed for several techniques developed in the approximation algorithms field. For instance, some of the earliest applications of the primal-dual method in this area have been to the SNDP problem. There was a sequence of papers applying this technique, eventually leading to the development of an $O(\log r_{\max})$-approximation algorithm [GGP$^+$94]. Subsequently, in the breakthrough work of [Jai01], Jain applied the technique of *iterative LP rounding* to this problem to obtain a substantially improved 2-approximation algorithm effectively settling the approximability of the problem (up to constant factors).

3.1 Problems Studied

In this thesis we extend the study of survivable network design problems in two different directions, the first result leading to the second.

Online SNDP. First, we study these problems in the *online* setting: we are given a graph with edge costs, and an upper bound $r_{\max}$ on the connectivity demand. Now a sequence of vertex pairs

$\{i,j\} \in V \times V$ is presented to us over time, each with some edge-connectivity demand r_{ij}—at this point we may need to buy some edges to ensure that all the edges bought by the algorithm provide an edge-connectivity of r_{ij} between vertices i and j. The goal is to remain competitive with the optimal offline solution of the current demand set, i.e., minimize the worst-case ratio of the cost of the online algorithm to that of the optimal solution (with the demand sequence known), where the worst-case is taken over all possible input demand sequences.

Stochastic SNDP. Secondly, we extend the online algorithm to the 2-stage stochastic version of the problem with independent demand arrivals, i.e., each pair of vertices $\{i,j\} \in V \times V$ has an associated probability p_{ij} of requiring r_{ij} edge-connectivity (i.e., the final solution must have r_{ij} edge-disjoint paths between i and j), where p_{ij} and r_{ij} are both given as input to our algorithm. As mentioned earlier, the algorithm can buy a subgraph H in the first stage, after which the exact set $\mathcal{D}$ of terminal pairs requiring connectivity is sampled. Then in the second stage, the algorithm must augment H with a subgraph $H_{\mathcal{D}}$ such that $H \cup H_{\mathcal{D}}$ has r_{ij} edge-disjoint paths for all $\{i,j\} \in \mathcal{D}$. The goal is to minimize $c(H) + \sigma \mathbb{E}_{\mathcal{D}}[c(H_{\mathcal{D}})]$

3.2 Our Results

Our first result is for the online version of the problem.

Theorem 3.1 *For the edge-connected survivable network design problem, there is an $\alpha = O(r_{\max} \log^3 n)$-competitive randomized online algorithm against oblivious adversaries.*

By combining this with a powerful tool known as *Boosted Sampling* [GPRS04], we can immediately get approximation algorithms with a similar ratio for the 2-stage stochastic version of the problem as well. This gives us the following result.

Theorem 3.2 *For the two-stage stochastic version of the edge-connected survivable network design problem (with independent demand arrivals), there is an $O(r_{\max} \log^3 n)$-approximation algorithm.*

Finally, for the special case when the input graph is complete, and the edge-costs satisfy the triangle inequality, we give a constant-factor approximation algorithm for the stochastic version of the problem.

Theorem 3.3 *For the two-stage stochastic version of the edge-connected survivable network design problem (with independent demand arrivals), there is a constant-factor approximation algorithm if the input graph is complete and the edge costs $c(\cdot)$ satisfy the triangle inequality.*

3.3 Techniques

An interesting aspect from a technical point of view is that all previous approaches (such as the primal-dual or iterative rounding techniques) for solving the SNDP problem use the global knowledge of the problem and seem to be inherently offline. Often this issue can be handled in the following manner: (i) solve the *fractional relaxation* of the problem online, and also perform an *online rounding* of the fractional solution. While the first part of this can in fact be handled using existing ideas, we are not aware of any rounding technique that would work in an "online" manner. This is because the rounding scheme most amenable to being performed online is independent randomized rounding, but such an approach fails even for the offline version of the Steiner forest problem.

Our approach to solve this problem is therefore to *reduce* the SNDP problem to an instance of the set cover problem, for which we know optimal online algorithms. However, naive ways of modeling SNDP as set cover involve constructing set systems with either exponentially many sets or elements, which would not give us any non-trivial competitive ratios. A somewhat surprising ingredient of our proof is that we use distance-preserving embeddings into random trees (i.e., into singly connected structures) to get algorithms for higher connectivity. In our work, these embeddings allow us to simplify the cost structure of the network and to abstract out a set cover instance which has size $n^{O(k)}$, and therefore enable us to obtain a competitive ratio guarantee of the form $k\text{polylog}(n)$. We should point out that the use of such randomized embeddings also implies that our algorithm is competitive only against oblivious adversaries that are not aware of the algorithm's random coin tosses. We now elaborate on our techniques in more detail.

3.3.1 Embedding into "Backboned Graphs"

One of the major advantages of network design problems which only sought 1-edge-connectivity is that one can embed the underlying metric space into random trees [Bar96, FRT04, EEST05, ABN08], where the problems are easier to (approximately) solve. Such a reduction seems impossible even for 2-edge-connectivity as the problem is trivially infeasible on a tree. However, the simple but crucial observation is to not ignore these ideas, as we show below.

Given a graph $G = (V, E)$ with edge lengths/costs $c(e)$, we probabilistically embed it into a *spanning* subtree (which we call the *base tree*) using the results of Elkin et al. and Abraham et al. [EEST05, ABN08]. Formally, this gives a random spanning tree $T = (V, E_T \subseteq E)$ of G with edge lengths $\widehat{c}_T$, such that for all $x, y \in V$:

1. $\widehat{c}_T(e) = c(e)$ for all edges $e \in E_T$, and hence $d_T(x, y) \geq d_G(x, y)$; and
2. $\mathbb{E}[d_T(x, y)] \leq \widetilde{O}(\log n) \cdot d_G(x, y)$, where d_G is the graph metric according to the edge lengths $c(e)$.

The distance d_T is defined in the obvious way: if $P_T(u, v)$ is the unique u-v path in T, then $d_T(u, v) = \sum_{e \in P_T(u,v)} \widehat{c}_T(e)$.

Constructing the Backboned Graph. Now *instead of throwing away non-tree edges,* imagine each non-tree edge $e = (u, v) \in E \setminus E_T$ being given a new weight $\widehat{c}_T(e) = \max\{c(e), d_T(u, v)\}$.

Note that the low-stretch embeddings of [EEST05, ABN08] probabilistically embed graphs into backboned graphs with a small expected stretch of $\widetilde{O}(\log n)$. Hence, for the subsequent steps in our algorithm (except those for the metric instances) we will assume that the input graph is a backboned graph, and will use its properties to design online algorithms and stochastic approximation algorithms.

3.3.2 A Compact Set Cover Instance for the Augmentation Problem

In this section we highlight how we can obtain significantly more structural properties about minimum cuts and covering them if the graph is backboned. In particular we show how we can augment connectivity (from, say, l to $l+1$) for a demand pair $\{s_i, t_i\}$ by showing that *all its minimal cuts* can be "covered" by a small collection of fundamental cycles of low cost. Note that naively casting this problem as an instance of set cover would either require exponentially many elements (i.e., the 2^n cuts separating s_i and t_i), or exponentially many sets (the "augmenting paths" between s_i and t_i), and in either case, the approximating guarantees of set cover would not result in non-trivial guarantees for our connectivity problem.

Let G be a backboned graph that is an instance of the SNDP problem with demand set $\mathcal{D}$, and let T be the base tree in G. For any edge $e = (u, v) \notin E_T$, define the *base cycle* O_e to be the fundamental cycle $\{e\} \cup P_T(u, v)$ of e with respect to T.

Characterizing the Set System. Let H be a subgraph which l-edge-connects (for some $l < k$) the vertices s_i and t_i for some demand pair $\{s_i, t_i\} \in \mathcal{D}$. Clearly the connectivity assumption implies that there are l edge-disjoint paths from s_i to t_i in H: denote this set of edge-disjoint paths by $\mathcal{P}_i$. Now if s_i and t_i are not $(l+1)$-edge-connected, then there exists at least one l-cut separating s_i and t_i, i.e., a set of l edges removing which would separate s_i and t_i in H. Furthermore, such l-cuts must pick *exactly one edge* from each path in $\mathcal{P}_i$. Let $\mathsf{Violated}_H(i)$ denote the set of all such such l-cuts.

Labeling. Consider any cut $Q \in \mathsf{Violated}_H(i)$. Since Q is a minimal l-cut for the demand pair s_i-t_i in H, it must be that any end vertex of a cut edge is reachable from one of s_i or t_i in $H \setminus Q$. We label each end vertex v reachable from s_i in $H \setminus Q$ by L (i.e., we set $\mathsf{label}(v) = L$), and each end vertex v reachable from t_i by R (we set $\mathsf{label}(v) = R$). We don't label the other vertices in $V(G)$.

Note that the labeling of the end vertices of a cut Q depends on the subgraph H and not just the set of edges in Q. However, given the subgraph H, any cut can be characterized by a collection of l edges, and a labeling in $\{L, R\}$ for each end-vertex of the l edges chosen, resulting in a total number of $O(m^l)$ cuts to cover[1].

We are now ready to present our main theorem which shows that we can "cover" all these cuts with a collection of sets of size at most m as well.

Theorem 3.4 (Cut Cover Theorem) *Consider an* SNDP *instance $\mathcal{I}$, and let* Opt *denote any optimal solution. Let $H \subseteq G$ be any subgraph that l-edge-connects terminal pair $\{s_i, t_i\}$ for some $l < k$, such that the base tree path $P_T(s_i, t_i) \subseteq H$. Then for any l-cut $Q \in \mathsf{Violated}_H(i)$, given the labeling of the endpoints of Q as described above, we can find an edge $e = (u, v) \in E(\mathsf{Opt})$ such that $O_e \setminus Q$ connects some L-vertex to some R-vertex. In particular, this ensures that s_i and t_i are connected in $(H \cup O_e) \setminus Q$ and so Q is no longer a cut separating s_i and t_i.*

The Set System. Putting this all together, we can consider the following set-system to for augmenting the edge-connectivity of a pair of vertices s_i and t_i from l to $(l+1)$ in a given subgraph H: each cut Q of l edges along with the corresponding labeling L or R (defined as above) is an "element"; each edge $e \in G$ is a "set"; and a set e covers an element Q if s_i and t_i are connected in the subgraph $H \cup O_e \setminus Q$.

Final Comments. It is easy to conclude from the above theorem that there is a feasible solution to the above instance of cost at most Opt, the cost of an optimal solution to the SNDP instance on the backboned graph. Since our reduction holds for any subgraph H, we can iteratively augment connectivity in this way to obtain an algorithm with non-trivial approximation guarantees.

[1] for an appropriate notion of "coverage"

Furthermore, since our set system is local (i.e., depends only on the given terminal pair s_i, t_i, and the current subgraph at hand), it easily extends to being solved online (and also easily extends to a cost-sharing scheme for the stochastic problem).

3.4 Related Work

Steiner network problems have received considerable attention in approximation algorithms: Agrawal et al. [AKR95] and Goemans and Williamson [GW95] used primal-dual methods to design approximation algorithms for Steiner forests and other 1-connectivity problems (and some higher connectivity problems where multiple copies of edges could be used). Klein and Ravi [KR93] gave an algorithm for the 2-connectivity problem, which was extended by Williamson et al. [WGMV95] and Goemans et al. [GGP$^+$94] to higher connectivity problems, yielding $O(\log k)$-approximation algorithms for k-connectivity, all using primal-dual methods. Jain [Jai01] gives an iterative rounding technique to obtain a 2-approximation algorithm for the most general problem of SNDP. These techniques have recently been employed to obtain tight results (assuming $P \neq NP$) for network design with degree constraints [LNSS07, LS08, BKN08]. Vertex connectivity problems are less well-understood: [CVV03, KN03, FL08] consider problems of *spanning* k-connectivity, and provide approximation algorithms with varying guarantees depending on k. Fleischer et al. [FJW06] give a 2-approximation for vertex connectivity when all $r_{ij} \in \{0, 1, 2\}$. Recently, improved approximation algorithms have been given for the problem of *single source* k-vertex connectivity [CCK08, CK08a], culminating in a simple greedy $O(k \log n)$ algorithm [CK08b]. In fact, the papers [CK08a, CK08b] also implicitly give $O(k)$-strict cost-shares for the single-source vertex-connectivity problem. As far as we can see, their techniques do not apply in the case of general survivable network design where vertex pairs do not share a common root, nor do they imply online algorithms with adversarial inputs. When the edges have metric costs, there are, quite expectedly, better approximation algorithms for vertex connectivity. Khuller and Raghavachari [KR96] gave $O(1)$-approximations for k-vertex-connected spanning subgraphs. Cheriyan and Vetta [CV05] later gave $O(1)$-approximations for the single source k-connected problem and a $O(\log r_{max})$-approximation for metric vertex-connected SNDP. Recently, Chan et al. [CFLY08] give constant factor approximations for several degree bounded problems on metric graphs. As for the inapproximability, Kortsarz et al. [KKL04] give $2^{\log^{1-\varepsilon} n}$ hardness results for the vertex-connected survivable network design problem.

Imase and Waxman [IW91] first considered the online Steiner tree problem and gave a tight $\Theta(\log |\mathcal{D}|)$-competitive algorithm. Awerbuch, Azar and Bartal [AAB04] generalized these results for the online Steiner forest problem, and subsequently Berman and Coulston [BC97] gave the same $\Theta(\log |\mathcal{D}|)$ guarantee. However, we do not see how to use these ideas for the general problem with higher connectivity. In fact, to the best of our knowledge, no online algorithms were previously known for this problem even for the online rooted 2-connectivity problem (i.e., for the case where all the vertex pairs share a root vertex r and the connectivity requirement is 2 for all pairs)—in fact, we also show a lower bound of $\Omega(\min\{|\mathcal{D}|, \log n\})$ on the competitive ratio for this special case, where $\mathcal{D}$ is the set of terminal pairs given to the algorithm. This is in contrast to the case of online 1-connectivity (i.e., online Steiner forest) where the best online algorithm is $\Theta(\log |\mathcal{D}|)$-competitive [BC97].

As mentioned before, we use the results of Alon et al. [AAA$^+$03] for the online (weighted) set cover problem; the ideas used in this paper have been extended by Alon et al. [AAA$^+$04] and Buchbinder and Naor [BN06] to get online primal-dual based algorithms for *fractional* generalized network design. We note that while we can solve the fractional version of the online k-connectivity problems using these techniques, we do not know how to round this fractional solution online.

As for the stochastic version of the problem, the only previous results known for these versions of higher-connectivity problems were $O(1)$-strict *cost-shares* implicitly given by Chuzhoy and Khanna [CK08b], and independently (but explicitly) by Chekuri et al. [CGK$^+$08] for the special case of *rooted* connectivity, where all pairs seek k-connectivity to a single source r (and hence to each other). The use of strict cost-shares to get algorithms for rent-or-buy network design appears in [GKPR07]. Approximation algorithms for two-stage stochastic problems were studied in [IKMM04a, RS04], and some general techniques were given by [GPRS04, SS06b]; in particular, using strict cost-shares to obtain approximation algorithms for stochastic optimization problems appears in [GPRS04].

4 Correlated Stochastic Knapsack and Multi-Armed Bandits

In this section, we consider adaptive stochastic scheduling problems where the input (i.e., the collection of job sizes and rewards) is distributional. More specifically, we consider the Stochastic Knapsack and the (more general) Multi-Armed Bandits problems and present constant-factor approximation algorithms for them. The Stochastic Knapsack problem has garnered much attention in the recent past both from the practical and theoretical points of view, starting with the work of Dean et al. [DGV05]. From the practical side, since they model the task of designing policies to schedule jobs only based on historic data about their runs, the common belief is that the algorithmic techniques developed could be useful in designing OS schedulers, real-time systems, etc. From the theoretical side, they introduce new challenges at all stages of algorithm design. Firstly, the optimal solution itself might require exponential size to describe (since it could be a complete decision tree, taking different actions for different random outcomes of the jobs). Secondly, our algorithm may also have to be adaptive (or else we need to bound the adaptivity gap to restrict our attention on designing non-adaptive solutions), and finally we would need to compare the expected profit of our (adaptive) algorithm with that of an optimal (adaptive) algorithm. We note that the disparities over set of issues to tackle means that solution techniques for solving these adaptive stochastic optimization problems have a very different theme than those for the two-stage stochastic problems.

The Multi-Armed Bandits problem is a classical problem which draws the interest of researchers in several areas of science (Statistics, Machine Learning, and of course, TCS) with research dating back to the 1950s. In the basic version of the MAB problem, there are a set of hidden distributions, where each distribution corresponds to the rewards delivered by one of the arms. The gambler (i.e., the algorithm) iteratively plays one arm per round and observes the associated reward. The objective is to maximize the sum of the collected rewards over a finite horizon of B plays. The most common metric for measuring the performance of an algorithm is known as *regret*. The regret ρ after B rounds is defined as the difference between the reward sum associated with an optimal strategy (which just involves playing the single arm with highest expected reward) and the sum of the collected rewards: $\rho = B\mu^* - \sum_{t=1}^{T} \widehat{r}_t$, where μ^* is the maximal reward mean and $\widehat{r}_t$ is the algorithm's reward at time t. A strategy whose average regret per round ρ/T tends to zero with probability 1 when the number of played rounds tends to infinity is known as a *no-regret* strategy. The version of the problem we study, however, is a bit different (it is more general in some ways, but the guarantees we seek are different).

4.1 Problems Studied

Correlated Stochastic Knapsack. In this problem, we are given a collection of jobs, each equipped with a distribution over ($\mathsf{size}, \mathsf{reward}$) pairs. The goal is to adaptively schedule these jobs so as to maximize the expected reward of all the jobs that have been successfully scheduled within a time budget B. The algorithm gets to know the actual size/reward of a job only when it completes, at which point it can take different decisions based on the outcomes of the previously scheduled jobs. We also study variants where the algorithm can prematurely cancel a job if it takes too long to complete.

Multi-Armed Bandits. We are given a collection of n arms; each arm corresponds to a Markov Chain (given as input), with each state being associated with some reward. At each timestep, the algorithm is given the current states that each Markov Chain is in, and it must decide which arm to play. The chosen arm then transitions into a new state as dictated by the Markov Chain, and the algorithm collects the reward from the new state. Our task is to design an (adaptive) algorithm which maximizes the total expected reward it can obtain over a horizon of B plays.

For both of the above problem, the benchmark for measuring the performance of an algorithm is the ratio $\mathbb{E}[\mathsf{Opt}]/\mathbb{E}[\mathsf{Alg}]$, where the individual expectations are over the randomness inherent in the problem (e.g., the random transitions of the Markov Chains) and possibly any randomization used by the algorithm.

A comparison to the **Min-Regret** *model.* The traditional model (adopted by the Statistics and Machine Learning communities) for measuring the performance of such scheduling algorithms, is the so-called *regret* model, where the goal is to minimize the *difference* between the reward of an optimal strategy and our algorithm's expected reward. On the one hand, it makes a finer measurement of performance since it uses an additive metric versus the multiplicative one we use. However, the model also compares our algorithm with the best optimal fixed strategy (which corresponds to the best fixed arm in hindsight) whereas we allow comparisons with an optimal adaptive algorithm. Finally, one other drawback of our model is that we work in the *full-information* setting, i.e., the individual Markov Chains are given as input, and the algorithm is aware of the states in which each arm is at all times. This differs from the regret-based models, where the algorithm also has to learn these parameters.

4.2 Our Results

We prove the following results in this thesis. Our first theorem concerns with the approximability (and adaptivity gap) of the basic stochastic knapsack problem.

Theorem 4.1 *There is a polynomial-time (non-adaptive) randomized algorithm for the stochastic knapsack problem with correlated rewards, which obtains an expected reward of at least $\frac{1}{8}\mathsf{Opt}$, where Opt is the expected reward of an optimal adaptive algorithm.*

We then extend this result to the model of the problem where the jobs can be prematurely canceled if they run for too long.

Theorem 4.2 *For the correlated stochastic knapsack problem with cancellations, there is a polynomial-time (non-adaptive) randomized algorithm which obtains an expected reward of at least $\frac{1}{16}\mathsf{Opt}$, where Opt is the expected reward of an optimal adaptive algorithm.*

Problem	Restrictions	Paper
Stochastic Knapsack	Sizes and Rewards Independent, No Cancellation	[DGV05]
	Sizes and Rewards Correlated, No Cancellation	This Thesis
	Sizes and Rewards Correlated, Cancellation	This Thesis
Multi-Armed Bandits	Martingale Property	[GM07b]
	No Martingale Property	This Thesis
MAB - Explore/Exploit	No Martingale Property	This Thesis

Table 1: Summary of Results

Finally, using the techniques we developed to solve the knapsack problem, we show the following result about the MAB problem.

Theorem 4.3 *There is a polynomial-time (adaptive) randomized algorithm for the* MAB *problem, which obtains an expected reward of at least* $\Omega(1)\mathsf{Opt}$*, where* Opt *is the expected reward of an optimal adaptive algorithm for the given* MAB *instance.*

We summarize the results in Table 1.

4.3 Techniques

One reason why stochastic packing problems are more difficult than deterministic ones is that, unlike in the deterministic setting, we cannot simply take a solution with expected reward R^* that packs into a knapsack of size $2B$ and get one with reward $\Omega(R^*)$ whilst fitting within the budget of B (by appropriately sub-selecting some items). In fact, in stochastic settings, there are examples where a budget of $2B$ can fetch much more reward than what a budget of size B can. Another distinction from deterministic problems is that allowing premature cancellations can drastically increase the solution value. The assumptions in previous works on stochastic knapsack and MAB avoided both issues, but we now need to tackle them.

Stochastic Knapsack: Dean et al. [DGV08, Dea05] assume that the reward of an item is independent of its size. Moreover, their model does not consider the possibility of canceling items in the middle. These assumptions simplify the structure of the optimal (adaptive) decision tree and make it possible to formulate a knapsack-style LP which captures Opt, and subsequently round it. However, their LP relaxation performs poorly when either correlation or cancellation is allowed.

Multi-Armed Bandits: Obtaining approximations for MAB problems is a more complicated task, since cancellations are inherent in the problem formulation (i.e., any strategy may stop playing a particular arm and switch to another) and the payoff of an arm is naturally correlated with its current state. While the first issue is tackled by using more elaborate LPs with a flow-like structure that compute a probability distribution over the different times at which the LP stops playing an arm (e.g., [GM07a]), the latter issue is less understood. Indeed, several papers on this topic present strategies that fetch an expected reward which is a constant-factor of an optimal solution's reward, but which may violate the budget by a constant factor. In order to obtain a good solution without violating the budget, they critically make use of the martingale property—with this assumption at hand, they can truncate the last arm played to fit the budget without incurring any loss in the expected reward. However, such an idea fails without the martingale property, and these LPs have large integrality gaps.

A major drawback with the previous LP relaxations is that the constraints are *local* for each item/arm, i.e., they track the probability distribution over how long each item/arm is processed, and there is a global constraint on the total number of pulls/knapsack budget. Using such local constraints results in two different issues.

For the (correlated) stochastic knapsack problem, these LPs do not capture the case when all the items have high *contention*, i.e., they may all want to be played early in order to collect a huge profit from their large sizes. And for the general multi-armed bandit problem, we show that *no such localized solution* can be good since they do not capture the notion of *preempting* an arm, namely switching from one arm to another and possibly returning to the original arm later. Indeed, we show cases when any near-optimal strategy must repeatedly switch back-and-forth between arms—this is the crucial difference from previous work with the martingale property where there exist near-optimal strategies that never return to any arm [GM09, Lemma 2.1]. Hence our algorithm needs to make *highly adaptive decisions*, contrasting with previously existing index-based policies.

We resolve these issues in the following manner. Incorporating item cancellations into stochastic knapsack can be done by adapting the flow-like LPs from earlier works on MABs. To handle the issues of contention and preemption, we formulate a *global time-indexed* relaxation that forces the LP solution to commit each item to begin at a time, and places constraints on the maximum expected reward that can be obtained if the LP begins an item at a particular time. Furthermore, the time-indexing also enables our rounding scheme to extract information about when to preempt an arm and when to re-visit it based on the LP solution; in fact, these decisions will depend on the (random) outcomes of previous pulls, but the LP encodes the information for each eventuality. We believe that our techniques are fairly general and would be applicable for other problems in Stochastic optimization.

4.4 Related Work

Stochastic scheduling problems (in fact, even those with correlated rewards) have been long studied since the 1960s (e.g., [BL97, Pin95, KP01, CFGW87]); however, there are fewer papers on approximation algorithms for such problems. These problems were first studied from an approximations perspective in an important paper of Dean et al. [DGV08] (see also [DGV05, Dea05]). They considered the *stochastic knapsack problem*, where each item has a random size and a random reward, and the sizes are revealed only after an item is placed into the knapsack; the goal is to give an adaptive strategy for picking items irrevocably in order to maximize the expected value of those fitting into a knapsack with size B. Via an LP relaxation and a rounding algorithm, they gave *non-adaptive* solutions with expected rewards that are (surprisingly) within a constant-factor of the best *adaptive* ones, resulting in a constant adaptivity gap (also a notion they introduced). However, the results required that (a) the random rewards and sizes are independent of each other, and (b) once an item was placed, it can not be prematurely canceled—it is easy to see that these assumptions change the nature of the problem significantly.

Kleinberg et al. [KRT00], and Goel and Indyk [GI99] consider stochastic knapsack problems with chance constraints: find the max-profit set which will overflow the knapsack with probability at most p. However, their results hold for deterministic profits and specific size distributions. The problem of minimizing average completion times with arbitrary job-size distributions was studied by [MSU99, SU01]. The work most relevant to us is that of Dean, Goemans and Vondrák [DGV08, DGV05, Dea05] on stochastic knapsack and packing; apart from algorithms (for independent rewards and sizes), they show the problem to be PSPACE-hard when correlations are allowed. [CR06] study stochastic flow problems. Recent work of Bhalgat et al. [BGK11] presents a PTAS but violate the capacity by a factor $(1+\varepsilon)$; they also get improved guarantees when there are no violations.

The general area of learning with costs is a rich and diverse one (see, e.g., [Ber05, Git89]). Variations of the MAB problem were introduced as early as in the 1950s [Rob52], with numerous applications in statistical estimation, resource allocation, etc. [GGW11]—however, most of the literature only address the infinite horizon models. The design of approximation algorithms for such problems starts with the work of Guha and Munagala [GM07a], who gave LP-rounding algorithms for some problems. In fact, the above developments in the stochastic knapsack problem have been crucial for understanding the multi-armed bandit problem (and other budgeted learning problems) from an approximation perspective. Indeed, recent works [GM07b, GM07a, GM09, GKN09, GMP11] have used ideas from [DGV08] to obtain the first multiplicative guarantees for several variants of the multi-armed bandit problems, but they all crucially rely on an additional assumption on the input, namely *the "martingale" property*: if an arm is some state u, one pull of this arm would bring an expected payoff equal to the payoff of state u itself.

Further papers by these authors [GMS07, GM09] and by Goel et al. [GKN09] give improvements, relate LP-based techniques and index-based policies and also give new index policies. (See also [GGM06, GM07b].) [GM09] considers switching costs, [GMP11] allows pulling many arms simultaneously, or when there is delayed feedback. All these papers assume the martingale condition.

5 Stochastic Orienteering

Our final focus in this thesis is on the orienteering problem with stochastic items located at nodes. For a practical motivation, consider the following problem: you start your day at home with a set of chores to run at various locations (e.g., at the bank, the post office, the grocery store), but you only have limited time to run those chores in (say, you have from 9am until 5pm, when all these shops close). Each successfully completed chore/job j gives you some fixed reward r_j. You know the time it takes you to travel between the various job locations: these distances are deterministic form a metric (V, d). However, you do not know the amount of time you will spend doing each job (e.g., standing in the queue, filling out forms). Instead, for each job j, you are only given the probability distribution π_j governing the random amount of time you need to spend performing j. That is, once you start performing the job j, the job finishes after size_j time units and you get the reward, where size_j is a random variable denoting the size, and distributed according to π_j.[2] (You may cancel processing the job j prematurely, but in that case you don't get any reward, and you are barred from trying j again.) The goal is now a natural one: given the metric (V, d), the starting point ρ, the time budget B, and the probability distributions for all the jobs, give a strategy for traveling around and doing the jobs that maximizes the expected reward accrued.

The case when all the sizes are deterministic (i.e., $\mathsf{size}_j = s_j$ with probability 1) is the orienteering problem, for which we now know a $(2+\varepsilon)$-approximation algorithm [BCK$^+$07, CKP08]. Another special case, where all the chores are located at the start node, but the sizes are random, is the stochastic knapsack problem, which also admits a $(2+\varepsilon)$-approximation algorithm [DGV08, Bha11]. However, the stochastic orienteering problem above, which combines aspects of both these problems,

[2]To clarify: before you reach the job, all you know about its size is what can be gleaned from the distribution π_j of size_j; and even having worked on j for t units of time, all you know about the actual size of j is what you can infer from the conditional $(\mathsf{size}_j \mid \mathsf{size}_j > t)$.

seems to have been hitherto unexplored in the approximation algorithms literature.

Just as in the previous section, our theoretical motivation stems from analyzing the "adaptivity gap" of this problem, and devising good *non-adaptive* solutions[3].

5.1 Our Results

In this thesis we show we can achieve the following approximations for the stochastic orienteering problem.

Theorem 5.1 *There is an $O(\log \log B)$-approximation algorithm for the stochastic orienteering problem.*

Indeed, our proof proceeds by first showing the following structure theorem which bounds the adaptivity gap:

Theorem 5.2 *Given an instance of the stochastic orienteering problem, then*

1. *either there exists a single job which gives an $\Omega(\log \log B)$ fraction of the optimal reward, or*
2. *there exists a value W^* such that the optimal non-adaptive tour which spends at most W^* time waiting and $B - W^*$ time traveling, gets an $\Omega(\log \log B)$ fraction of the optimal reward.*

Note that naïvely we would expect only a logarithmic fraction of the reward, but the structure theorem shows we can do better. Indeed, this theorem is the technical heart of the analysis, and is proved via a martingale argument, that we believe could be of independent interest. Since the above theorem shows the existence of a non-adaptive solution close to the best adaptive solution, we can combine it with the following result to prove Theorem 5.1.

Theorem 5.3 *There exists a constant-factor approximation algorithm to the optimal non-adaptive policy for stochastic orienteering.*

Note that if we could show an existential proof of a constant adaptivity gap (which we conjecture to be true), the above approximation for non-adaptive problems that we show immediately implies an $O(1)$-approximation algorithm for the adaptive problem too.

5.2 Techniques

A natural idea for StocOrient is to replace random jobs by deterministic ones with size equal to the expected size $E[S_v]$, find a near-optimal orienteering solution P to the deterministic instance which gets reward R. One can then use this path P to get a non-adaptive strategy for the original StocOrient instance with expected reward $\Omega(R)$. Indeed, suppose the path P spends time L traveling and W waiting on the deterministic jobs such that $L + W \leq B$. Then, picking a random half of the jobs and visiting them results in a non-adaptive solution for StocOrient which travels at most L and processes jobs for time at most $W/2$ in expectation. Hence, Markov's inequality says that

[3] A non-adaptive solution for stochastic orienteering is simply a tour P of points in the metric space starting at the root ρ: we visit the points in this fixed order, performing the jobs at the points we reach, until time runs out

with probability at least $1/2$, all jobs finish processing within W time units and we get the entire reward of this sub-path, which is $\Omega(R)$.

However, the problem is in showing that $R = \Omega(\mathsf{Opt})$—i.e., that the deterministic instance has a solution with large reward. The above simplistic reduction of replacing random jobs by deterministic ones with mean size fails even for stochastic knapsack: suppose the knapsack budget is B, and each of the n jobs have size Bn with probability $1/n$, and size 0 otherwise. Note that the expected size of every job is now B. Therefore, a deterministic solution can pick only one job, whereas the optimal solution would finish $\Omega(n)$ jobs with high probability. However, observe that this problem disappears if we truncate all sizes at the budget, i.e., set the deterministic size to be the expected "truncated" size $E[\min(S_j, B)]$ where S_j is the random size of job j. (Of course, we also have to set the reward to be $r_j\,\mathbb{P}[\mathsf{size}_j \leq B]$ to discount the reward from impossible size realizations.) Now $\mathbb{E}[\min(W_j, B)]$ reduces to B/n and so the deterministic instance can now get $\Omega(n)$ reward. Indeed, this is the approach used by [DGV08] to get an $O(1)$-approximation and adaptivity gap.

But for $\mathsf{StocOrient}$, is there a good truncation threshold? Considering $\mathbb{E}[\min(\mathsf{size}_j, B)]$ fails on the example where all jobs are co-located at a point at distance $B-1$ from the root. Each job v has size B with probability $1/B$, and 0 otherwise. Truncation by B gives an expected size $\mathbb{E}_{\mathsf{size}_v \sim \pi_v}[\min(\mathsf{size}_v, B)] = 1$ for every job, and so the deterministic instance gets reward from only one job, while the $\mathsf{StocOrient}$ optimum can collect $\Omega(B)$ jobs. Now noticing that any algorithm *has* to spend $B-1$ time traveling to reach any vertex that has some job, we can instead truncate each job j size at $B - d(\rho, j)$, which is the maximum amount of time we can possibly spend at j (since we must reach vertex j from ρ). However, while this fix would work for the aforementioned example, there are examples which show that the deterministic instance gets only an $O(\frac{\log\log B}{\log B})$ fraction of the optimal stochastic reward.

However, suppose we replace each job by its expected size $\mu_i = B/(2^i \log B)$, and try to pick jobs so that the total travel plus expected sizes is at most B. Suppose j is the first job we pick along the line, then because of its size being μ_j we cannot reach any jobs in the last μ_j length of the path. The number of these lost jobs is $\log \mu_j = \log B - j - \log\log B$ (because of the geometrically decreasing gaps between jobs), and hence we can reach only jobs $j, j+1, j+\log\log B - 1$—giving us a maximum profit of $\log\log B$ even if we ignore the space these jobs would take. (In fact, since their sizes decrease geometrically, we can indeed get all but a constant number of these jobs.)

This shows that replacing jobs in a $\mathsf{StocOrient}$ instance by their expected truncated sizes gives a deterministic instance whose optimal reward is smaller by an $\Omega(\frac{\log B}{\log\log B})$ factor.

Why the above approaches failed, and our approach. The reason why the determinization techniques described above worked for stochastic knapsack, but failed for stochastic orienteering is the following: the total sizes of jobs is always roughly B in knapsack (so truncating at B was the right thing to do). But in orienteering it depends on the total time spent traveling, *which in itself is a random quantity even for a non-adaptive solution.* One way around this is to guess the amount of time spent processing jobs (up to a factor of 2) which gets the largest profit, and use that as the truncation threshold. Such an approach seems like it should lose a $\Omega(\log B)$ fraction of the reward. *Surprisingly, we show that this simple algorithm actually gives a much better reward: it gets a constant factor of the reward of a non-adaptive optimum, and an $O(\log\log B)$-approximation when compared to the adaptive optimum!*

At a high level, our approach is to solve this problem in three steps. Suppose we start off with an instance $\mathcal{I}_{so}$ of $\mathsf{StocOrient}$ with optimal (adaptive) solution having expected reward Opt. The high level overview of our algorithm is described below:

Step 1: Construct a suitable instance $\mathcal{I}_{ko}$ of *Knapsack Orienteering* ($\mathsf{KnapOrient}$), with the guarantee that the optimal profit from this $\mathsf{KnapOrient}$ instance $\mathcal{I}_{ko}$ is $\Omega(\mathsf{Opt}/\log\log B)$.

Step 2: Find a path P with reward $\Omega(\mathsf{Opt}/\log\log B)$ for $\mathcal{I}_{ko}$.

Step 3: Follow this solution P and process the jobs in the order P visits them.

6 Future Work

In this section, we discuss the problems which we intend to work on during the coming six months, before May 2012 when I plan to complete this thesis. A few of these problems require substantially different techniques to solve them while some others appear easier to handle.

1. *Stochastic Vertex*-SNDP. The main question is how to use the techniques we design to solve the vertex-connectivity problem, i.e., requiring vertex disjoint paths. The main hurdle here is that we don't have a clean characterization of the "cuts" and "coverings" as we do for the case of edge-connectivity. A good starting-step is to handle just the case of *rooted* 2-*vertex-connected subgraphs*, in either the online or stochastic setting.

2. *Stochastic Routing Problems.* This is a more general modeling question. I would like to understand if there is a nice formulation of studying the 2-stage stochastic optimization problem for multi-commodity flow problems on capacitated graphs. One such example could be that buying some capacity up front has a cost function associated, and subsequently (after the actual demands are revealed) augmenting the connectivity has a different cost function. Can we provide approximation results for such problems?

3. *Adaptivity Gap in Stochastic Orienteering.* In this problem, the main avenue of future work would be to get a complete understanding of the adaptivity gap of the basic problem. Our current bound is $O(\log\log B)$, while we believe the right answer is $O(1)$. We would like to explore more and see if our analysis could be tightened to improve the gap to a constant-factor.

4. *Variants of* StocOrient. In addition to the basic model of StocOrient we consider, we could also look at variants where the graph is directed, or the model where the algorithm needs to end up at a specified end-vertex before the budget runs out, etc. Some of these could be simple extensions, and other variants might require some new ideas.

Acknowledgements. The results on Stochastic Survivable Network Design are based on joint work with Anupam Gupta and R. Ravi and appear in [GKR09]. The results on the Stochastic Knapsack and MAB problems are based on joint work with Anupam Gupta, Marco Molinaro, and R. Ravi and appear in [GKMR11], and finally those on the orienteering problem are based on collaboration [GKNR12] with Anupam Gupta, Viswanath Nagarajan, and R. Ravi. I would like to thank the aforementioned co-authors for graciously allowing me to include the results in this thesis.

References

[AAA+03] Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Seffi Naor. The online set cover problem. In *35th STOC*, pages 100–105, 2003.

[AAA$^+$04] Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Seffi Naor. A general approach to online network optimization problems. In *15th SODA*, pages 577–586, 2004.

[AAB04] Baruch Awerbuch, Yossi Azar, and Yair Bartal. On-line generalized Steiner problem. *Theoret. Comput. Sci.*, 324(2-3):313–324, 2004.

[ABN08] Ittai Abraham, Yair Bartal, and Ofer Neiman. Nearly tight low stretch spanning trees. In *48th FOCS*, pages 781–790, 2008.

[AKR95] Ajit Agrawal, Philip Klein, and R. Ravi. When trees collide: an approximation algorithm for the generalized Steiner problem on networks. *SIAM J. Comput.*, 24(3):440–456, 1995. (Preliminary version in *23rd STOC*, 1991).

[Bar96] Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th FOCS*, pages 184–193, 1996.

[BC97] Piotr Berman and Chris Coulston. On-line algorithms for steiner tree problems (extended abstract). In *29th STOC*, pages 344–353, 1997.

[BCK$^+$07] Avrim Blum, Shuchi Chawla, David R. Karger, Terran Lane, Adam Meyerson, and Maria Minkoff. Approximation algorithms for orienteering and discounted-reward TSP. *SIAM J. Comput.*, 37(2):653–670 (electronic), 2007.

[Bea55] E. M. L. Beale. On minimizing a convex function subject to linear inequalities. *Journal of the Royal Statistical Society. Series B (Methodological)*, 17(2):pp. 173–184, 1955.

[Ber05] Dimitri P. Bertsekas. *Dynamic programming and optimal control.* Athena Scientific, Belmont, MA, third edition, 2005.

[BGK11] Anand Bhalgat, Ashish Goel, and Sanjeev Khanna. Improved approximation results for stochastic knapsack problems. In *SODA '11*. Society for Industrial and Applied Mathematics, 2011.

[Bha11] Anand Bhalgat. A $(2 + \varepsilon)$-approximation algorithm for the stochastic knapsack problem. 2011.

[BKN08] Nikhil Bansal, Rohit Khandekar, and Viswanath Nagarajan. Additive guarantees for degree bounded directed network design. In *40th STOC*, pages 769–778, 2008.

[BL97] John R. Birge and François Louveaux. *Introduction to stochastic programming.* Springer Series in Operations Research. Springer-Verlag, 1997.

[BN06] Niv Buchbinder and Joseph (Seffi) Naor. Improved bounds for online routing and packing via a primal-dual approach. In *46th FOCS*, pages 293–304, 2006.

[CCK08] Tanmoy Chakraborty, Julia Chuzhoy, and Sanjeev Khanna. Network design for vertex connectivity. In *40th STOC*, pages 167–176, 2008.

[CFGW87] Jr. Coffman, E. G., L. Flatto, M. R. Garey, and R. R. Weber. Minimizing expected makespans on uniform processor systems. *Adv. Appl. Prob.*, 19(1):pp. 177–201, 1987.

[CFLY08] Yuk Hei Chan, Wai Shing Fung, Lap Chi Lau, and Chun Kong Yung. Degree bounded network design with metric costs. In *48th FOCS*, 2008.

[CGK+08] Chandra Chekuri, Anupam Gupta, Nitish Korula, Ravishankar Krishnaswamy, and Amit Kumar. Cost-sharing for subgraph k-edge-connectivity. unpublished, July 2008.

[CK08a] Chandra Chekuri and Nitish Korula. Single-sink network design with vertex connectivity requirements. In *FST&TCS*, 2008.

[CK08b] Julia Chuzhoy and Sanjeev Khanna. Algorithms for single-source vertex connectivity. In *48th FOCS*, pages 105–114, 2008.

[CKP08] Chandra Chekuri, Nitish Korula, and Martin Pál. Improved algorithms for orienteering and related problems. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 661–670, New York, 2008. ACM.

[CR06] Shuchi Chawla and Tim Roughgarden. Single-source stochastic routing. In *Proceedings of APPROX*, pages 82–94. 2006.

[CV05] Joseph Cheriyan and Adrian Vetta. Approximation algorithms for network design with metric costs. In *37th STOC*, pages 167–175, 2005.

[CVV03] Joseph Cheriyan, Santosh Vempala, and Adrian Vetta. An approximation algorithm for the minimum-cost k-vertex connected subgraph. *SIAM J. Comput.*, 32(4):1050–1055, 2003.

[Dan04] George B. Dantzig. Linear programming under uncertainty. *Manage. Sci.*, 50:1764–1769, December 2004.

[Dea05] Brian C. Dean. *Approximation Algorithms for Stochastic Scheduling Problems.* PhD thesis, MIT, 2005.

[DGV05] Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Adaptivity and approximation for stochastic packing problems. In *SODA*, pages 395–404, 2005.

[DGV08] Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.*, 33(4):945–964, 2008.

[DST03] Shane Dye, Leen Stougie, and Asgeir Tomasgard. The stochastic single resource service-provision problem. *Naval Research Logistics (NRL)*, 50(8):869–887, 2003.

[EEST05] Michael Elkin, Yuval Emek, Daniel A. Spielman, and Shang-Hua Teng. Lower-stretch spanning trees. In *37th STOC*, pages 494–503, 2005.

[FJW06] Lisa Fleischer, Kamal Jain, and David P. Williamson. Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *J. Comput. System Sci.*, 72(5):838–867, 2006.

[FL08] Jittat Fakcharoenphol and Bundit Laekhanukit. An O($\log^2$ k)-approximation algorithm for the k-vertex connected spanning subgraph problem. In *40th STOC*, pages 153–158, 2008.

[FRT04] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. System Sci.*, 69(3):485–497, 2004.

[GGM06] Ashish Goel, Sudipto Guha, and Kamesh Munagala. Asking the right questions: model-driven optimization using probes. In *PODS*, pages 203–212, 2006.

[GGP+94] Michel X. Goemans, Andrew V. Goldberg, Serge Plotkin, David B. Shmoys, Éva Tardos, and David P. Williamson. Improved approximation algorithms for network design problems. In *5th SODA*, pages 223–232, 1994.

[GGW11] John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-armed Bandit Allocation Indices*. Wiley Interscience, 2011.

[GI99] Ashish Goel and Piotr Indyk. Stochastic load balancing and related problems. In *40th Annual Symposium on Foundations of Computer Science (New York, 1999)*, pages 579–586. IEEE Computer Soc., Los Alamitos, CA, 1999.

[Git89] J. C. Gittins. *Multi-armed bandit allocation indices*. Wiley-Interscience Series in Systems and Optimization. John Wiley & Sons Ltd., Chichester, 1989. With a foreword by Peter Whittle.

[GK11] Anupam Gupta and Jochen Konemann. Approximation algorithms for network design: A survey. *Surveys in Operations Research and Management Science*, 16(1):3 – 20, 2011.

[GKMR11] Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. Approximation algorithms for correlated knapsack and non-martingale bandits. In *51st STOC*, 2011.

[GKN09] Ashish Goel, Sanjeev Khanna, and Brad Null. The ratio index for budgeted learning, with applications. In *SODA '09: Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 18–27, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[GKNR12] Anupam Gupta, Ravishankar Krishnaswamy, Viswanath Nagarajan, and R. Ravi. Approximation algorithms for stochastic orienteering. In *23rd SODA*, 2012.

[GKPR07] Anupam Gupta, Amit Kumar, Martin Pál, and Tim Roughgarden. Approximation via cost sharing: simpler and better approximation algorithms for network design. *J. ACM*, 54(3):Art. 11, 38 pp. (electronic), 2007.

[GKR09] Anupam Gupta, Ravishankar Krishnaswamy, and R. Ravi. Online and stochastic survivable network design. In *41st STOC*, 2009.

[GM07a] Sudipto Guha and Kamesh Munagala. Approximation algorithms for budgeted learning problems. In *STOC'07—Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 104–113. ACM, New York, 2007.

[GM07b] Sudipto Guha and Kamesh Munagala. Model-driven optimization using adaptive probes. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 308–317, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[GM09] Sudipto Guha and Kamesh Munagala. Multi-armed bandits with metric switching costs. In *ICALP*, pages 496–507, 2009.

[GMP11] Sudipto Guha, Kamesh Munagala, and Martin Pal. Iterated allocations with delayed feedback. *ArXiv*, arxiv:abs/1011.1161, 2011.

[GMS07] Sudipto Guha, Kamesh Munagala, and Peng Shi. On index policies for restless bandit problems. *CoRR*, abs/0711.3861, 2007.

[GPRS04] Anupam Gupta, Martin Pál, R. Ravi, and Amitabh Sinha. Boosted sampling: Approximation algorithms for stochastic optimization problems. In *36th STOC*, pages 417–426, 2004.

[GW95] Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.

[IKMM04a] Nicole Immorlica, David Karger, Maria Minkoff, and Vahab Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *15th SODA*, pages 684–693, 2004.

[IKMM04b] Nicole Immorlica, David Karger, Maria Minkoff, and Vahab S. Mirrokni. On the costs and benefits of procrastination: approximation algorithms for stochastic combinatorial optimization problems. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '04, pages 691–700, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[IW91] Makoto Imase and Bernard M. Waxman. Dynamic Steiner tree problem. *SIAM J. Discrete Math.*, 4(3):369–384, 1991.

[Jai01] Kamal Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001. (Preliminary version in *39th FOCS*, pages 448–457, 1998).

[KKL04] Guy Kortsarz, Robert Krauthgamer, and James R. Lee. Hardness of approximation for vertex-connectivity network design problems. *SIAM J. Comput.*, 33(3):704–720, 2004.

[KN03] Guy Kortsarz and Zeev Nutov. Approximating node connectivity problems via set covers. *Algorithmica*, 37(2):75–92, 2003.

[KP01] Anton J. Kleywegt and Jason D. Papastavrou. The dynamic and stochastic knapsack problem with random sized items. *Oper. Res.*, 49:26–41, January 2001.

[KR93] Philip N. Klein and R. Ravi. When cycles collapse: A general approximation technique for constrained two-connectivity problems. In *3rd IPCO*, pages 39–56, 1993.

[KR96] Samir Khuller and Balaji Raghavachari. Improved approximation algorithms for uniform connectivity problems. *J. Algorithms*, 21(2):434–450, 1996.

[KRT00] Jon Kleinberg, Yuval Rabani, and Éva Tardos. Allocating bandwidth for bursty connections. *SIAM J. Comput.*, 30(1):191–217, 2000.

[LNSS07] Lap Chi Lau, Joseph Naor, Mohammad R. Salavatipour, and Mohit Singh. Survivable network design with degree or order constraints. In *39th STOC*, pages 651–660, 2007.

[LS08] Lap Chi Lau and Mohit Singh. Additive approximation for bounded degree survivable network design. In *40th STOC*, pages 759–768, 2008.

[MSU99] Rolf H. Möhring, Andreas S. Schulz, and Marc Uetz. Approximation in stochastic scheduling: the power of LP-based priority policies. *Journal of the ACM (JACM)*, 46(6):924–942, 1999.

[Pin95] Michael Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Prentice Hall, 1995.

[Rob52] Herbert Robbins. Some aspects of the sequential design of experiments. *Bull. AMS*, 58(5):527–535, 1952.

[RS04] R. Ravi and Amitabh Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. In *10th IPCO*, pages 101–115, 2004.

[RS06] R. Ravi and Amitabh Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. *Math. Program.*, 108:97–114, August 2006.

[SS06a] David B. Shmoys and Chaitanya Swamy. An approximation scheme for stochastic linear programming and its application to stochastic integer programs. *J. ACM*, 53:978–1012, November 2006.

[SS06b] David B. Shmoys and Chaitanya Swamy. An approximation scheme for stochastic linear programming and its application to stochastic integer programs. *J. ACM*, 53(6):978–1012, 2006.

[SU01] Martin Skutella and Marc Uetz. Scheduling precedence-constrained jobs with stochastic processing times on parallel machines. In *12th SODA*, pages 589–590. Society for Industrial and Applied Mathematics, 2001.

[WGMV95] David P. Williamson, Michel X. Goemans, Milena Mihail, and Vijay V. Vazirani. A primal-dual approximation algorithm for generalized Steiner network problems. *Combinatorica*, 15(3):435–454, 1995.