Stochastic Combinatorial Optimization via Poisson Approximation

Jian Li
IIIS, Tsinghua University
Beijing, P.R.China
lijian83@mail.tsinghua.edu.cn

Wen Yuan
IIIS, Tsinghua University
Beijing, P.R.China
yuan-w10@mails.tsinghua.edu.cn

ABSTRACT

We study several stochastic combinatorial problems, including the expected utility maximization problem, the stochastic knapsack problem and the stochastic bin packing problem. A common technical challenge in these problems is to optimize some function (other than the expectation) of the sum of a set of random variables. The difficulty is mainly due to the fact that the probability distribution of the sum is the convolution of a set of distributions, which is not an easy objective function to work with. To tackle this difficulty, we introduce the Poisson approximation technique. The technique is based on the Poisson approximation theorem discovered by Le Cam, which enables us to approximate the distribution of the sum of a set of random variables using a compound Poisson distribution. Using the technique, we can reduce a variety of stochastic problems to the corresponding deterministic multiple-objective problems, which either can be solved by standard dynamic programming or have known solutions in the literature. For the problems mentioned above, we obtain the following results:

1. We first study the expected utility maximization problem introduced recently [Li and Despande, FOCS11]. For monotone and Lipschitz utility functions, we obtain an additive PTAS if there is a multidimensional PTAS for the multi-objective version of the problem, strictly generalizing the previous result. The result implies the first additive PTAS for maximizing threshold probability for the stochastic versions of global min-cut, matroid base and matroid intersection.
2. For the stochastic bin packing problem (introduced in [Kleinberg, Rabani and Tardos, STOC97]), we show there is a polynomial time algorithm which uses at most the optimal number of bins, if we relax the size of each bin and the overflow probability by ϵ for any constant $\epsilon > 0$. Based on this result, we obtain a 3-approximation if only the size of each bin can be relaxed by ϵ, improving the known $O(1/\epsilon)$ factor for constant overflow probability.
3. For stochastic knapsack, we show a $(1+\epsilon)$-approximation using ϵ extra capacity for any $\epsilon > 0$, even when the size and reward of each item may be correlated and cancelations of items are allowed. This generalizes the previous work [Balghat, Goel and Khanna, SODA11] for the case without correlation and cancelation. Our algorithm is also simpler. We also present a factor $2+\epsilon$ approximation algorithm for stochastic knapsack with cancelations, for any constant $\epsilon > 0$, improving the current known approximation factor of 8 [Gupta, Krishnaswamy, Molinaro and Ravi, FOCS11].
4. We also study an interesting variant of the stochastic knapsack problem, where the size and the profit of each item are revealed before the decision is made. The problem falls into the framework of Bayesian online selection problems, which has been studied a lot recently.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
STOC'13, June 1ÍC4, 2013, Palo Alto, California, USA.
Copyright 2013 ACM 978-1-4503-2029-0/13/06 ...$15.00.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: [Non-numerical Algorithms and Problems]

General Terms

Theory, Algorithms

Keywords

Stochastic Knapsack; Stochastic Bin Packing; Expected Utility Maximization

1. INTRODUCTION

We study several stochastic combinatorial optimization problems, including the threshold probability maximization problem [37, 35, 32], the expected utility maximization problem [32], the stochastic knapsack problem [18, 9, 23], the stochastic bin packing problem [28, 21] and some of their variants. All of these problems are known to be #P-hard and we are interested in obtaining approximation algorithms with provable performance guarantees. We observe a common technical challenge in solving these problems, that is, roughly speaking, given a set of random variables with possibly different probability distributions, to find a subset of random variables such that certain functional (other than the expectation) of their sum is optimized. The difficulty is mainly attributed to the fact that the probability distribution of the sum is the convolution of the distributions of individual random variables. To address this issue, we introduce a new technique, called the *Poisson approximation technique*, which can be used to approximate the probability distribution of a sum of several random variables. The technique is quite easy to use and yields better or more general results than the previous techniques for a variety of stochastic combinatorial optimization problems mentioned above.

In the rest of the section, we formally introduce these problems and state our results.

Terminology: We first set up some notations and review some standard terminologies. Following the literature, the *exact version* of a problem $\mathfrak{A}$ (denoted as Exact-$\mathfrak{A}$) asks for a feasible solution of $\mathfrak{A}$ with weight exactly equal to a given number K. An algorithm runs in *pseudopolynomial time* for Exact-$\mathfrak{A}$ if the running time of the algorithm is bounded by a polynomial of n and K. In a *multidimensional minimization* problem, each element b is associated with a weight vector $(w_{b,1}, w_{b,2}, \ldots)$. We are also given a budget vector $(B_1, B_2, \ldots)$. The goal is to find a feasible solution $S \in \mathcal{F}$ such that $\sum_{b\in S} w_{b,i} \leq B_i \, \forall i$. We use Multi-$\mathfrak{A}$ to denote the problem if the corresponding single dimensional optimization problem is $\mathfrak{A}$. A *multidimensional PTAS* for Multi is an algorithm which either returns a feasible solution $S \in \mathcal{F}$ such that $\sum_{b\in S} w_{b,i} \leq (1+\epsilon)B_i \, \forall i$, or asserts that there is no solution S' with $\sum_{b\in S'} w_{b,i} \leq B_i \, \forall i$.

1.1 Expected Utility Maximization

We first consider the fixed set model of a class of stochastic optimization problems introduced in [32]. We are given a ground set of elements (or items) $\mathbb{B} = \{b_i\}_{i=1\ldots n}$. Each feasible solution to the problem is a subset of the elements satisfying some property. In the deterministic version of the problem, we want to find a feasible solution S with the minimum total weight. In the stochastic version, each element b is associated with a random weight X_b. We assume all X_bs are discrete nonnegative random variables and are independent of each other. We are also given a utility function $\mu : \mathbb{R} \to \mathbb{R}^+$ to capture different *risk-averse* or *risk-prone* behaviors that are commonly observed in decision-making under uncertainty. Our goal is to to find a feasible set S such that the expected utility $\mathbb{E}[\mu(X(S))]$ is maximized. We refer to this problem as the *expected utility maximization (EUM)* problem. An important special case is to find a feasible set S such that $\Pr[X(S) \leq 1]$ is maximized, which we call the *threshold probability maximization (TPM)* problem. In fact, this special case has been studied extensive in literature for various combinatorial problems including stochastic versions of shortest path [37], minimum spanning tree [26, 20], knapsack [21] as well as some other problems [35]. We use $\mathfrak{A}$ to denote the deterministic version of the optimization problem under consideration, and use EUM-$\mathfrak{A}$ and TPM-$\mathfrak{A}$ to denote the expected utility maximization problem and the threshold probability maximization problem for $\mathfrak{A}$ respectively.

Our Results: Following the previous work [32], we assume $\lim_{x\to\infty} \mu(x) = 0$ (if the weight of our solution is too large, it is almost useless). We also assume μ is α-Lipschitz in $[0, \infty)$, i.e., $|\mu(x) - \mu(y)| \leq \alpha|x - y|$ for any $x, y \in [0, \infty)$, where α is a positive constant. Our first result is an alternative proof for the main result in [32].

THEOREM 1.1. *Assume there is a pseudopolynomial time algorithm for Exact-$\mathfrak{A}$. For any $\epsilon > 0$, there is a poly-time approximation algorithm for EUM-$\mathfrak{A}$ that finds a feasible solution S such that $\mathbb{E}[\mu(X(S))] \geq \mathsf{OPT} - \epsilon$.*

For many combinatorial problems, including shortest path, spanning tree, matching and knapsack, a pseudopolynomial algorithm for the exact version is known. Therefore, Theorem 1.1 immediately implies an additive PTAS for the EUM version of each of these problems. An important corollary of the above theorem is a *relaxed* additive PTAS for TPM: For any $\epsilon > 0$, we can find in polynomial time a feasible solution S such that $\Pr[X(S) \leq 1 + \epsilon] \geq \mathsf{OPT} - \epsilon$, provided that there is a pseudopolynomial time algorithm for Exact-$\mathfrak{A}$. We refer the interested reader to [32] for more implications of Theorem 1.1. However, this is not the end of story. Our second major result considers EUM with monotone nonincreasing utility functions, a natural class of utility functions (we denoted the problem as EUM-Mono). We can get the following strictly more general result for EUM-Mono.

THEOREM 1.2. *Assume there is a multidimensional PTAS for Multi-$\mathfrak{A}$. For any $\epsilon > 0$, there is a poly-time approximation algorithm for EUM-Mono-$\mathfrak{A}$ that finds a feasible solution S such that $\mathbb{E}[\mu(X(S))] \geq \mathsf{OPT} - \epsilon$.*

It is worthwhile mentioning that condition of Theorem 1.2 is strictly more general than the condition of Theorem 1.1. It is known that if there is pseudopolynomial time algorithm for Exact-$\mathfrak{A}$, there is a multidimensional PTAS for Multi-$\mathfrak{A}$, by Papadimitriou and Yannakakis [38]. However, the converse is not true. Consider the minimum cut (MC) problem. A pseudopolynomial time algorithm for Exact-MC would imply a polynomial time algorithm for the NP-hard MAX-CUT problem, while a multidimensional PTAS for Multi-MC is known [2]. Therefore, Theorem 1.2 implies the first relaxed additive PTAS for TPM-MC. Other problems that can justify the superiority of Theorem 1.2 include the weighted matroid base problem and the weighted matroid intersection problem (with some cardinality lower bound), for which the existence of any pseudopolynomial time exact algorithms is still open [10], while multidimensional PTASes are known[39, 15]. Lastly, we remark that obtaining an additive PTAS for EUM-$\mathfrak{A}$ for non-monotone utility functions under the same condition as Theorem 1.2 is impossible. The details can be found in the full version of the paper.

1.2 Stochastic Bin Packing

In the stochastic bin packing (SBP) problem, we are given a set of items $\mathbb{B} = \{b_1, b_2, \ldots, b_n\}$ and an overflow probability $0 < p < 1$. The size of each item b_i is an independent random variable X_i following a known discrete distribution. The distributions for different items may be different. Each bin has a capacity of $\mathbb{C}$. The goal is to pack all the items in $\mathbb{B}$ using as few bins as possible such that the overflow probability for each bin is at most p. The problem was first studied by Kleinberg, Rabani and Tardos [28]. They obtained a $O(1/\epsilon)$-approximation, for only Bernoulli distributed items, if we relax the bin size to $(1+\epsilon)\mathbb{C}$ or the overflow probability to $p^{1-\epsilon}$. They also obtained a $O(\sqrt{\frac{\log p^{-1}}{\log\log p^{-1}}})$-approximation without relaxing the bin size and the overflow probability. Goel and Indyk [21] obtained PTAS for both Poisson and exponential distributions and QPTAS (i.e., quasi-polynomial) for Bernoulli distribution.

Our Results: Our main result is the following theorem.

THEOREM 1.3. *For any fixed constant $\epsilon > 0$, there is a polynomial time algorithm for SBP that uses at most the optimal number of bins, when the bin size is relaxed to $(1+\epsilon)\mathbb{C}$ and the overflow probability is relaxed to $p + \epsilon$.*

To the best of our knowledge, our result is the first result for SBP for arbitrary distributions. Based on this result,we can get a 3-approximation when the overflow probability is not relaxed. For Bernoulli distributions, this improves the $O(1/\epsilon)$-approximation in [28] for any constant p.

THEOREM 1.4. *For any constant $p > 0$, we can find in polynomial time a packing that uses at most* 3OPT *bins of capacity $(1+\epsilon)\mathbb{C}$ such that the overflow probability of each bin is at most p.*

1.3 Stochastic Knapsack

In the stochastic knapsack problem, the size and/or the profit of an item may not be fixed values and only their probability distributions are known to us in advance. The actual size and profit of an item/job are revealed to us as soon as it is inserted into the knapsack with capacity $\mathbb{C}$. If the insertion of an item causes the knapsack to overflow, we terminate and do not gain the profit of that item. Unlike the deterministic knapsack problem for which a solution is a subset of items, a solution to SK is specified by an *adaptive policy* which determines which item to insert next based on the remaining capacity and the set of available items.

A significant generalization of the problem, introduced in [23], considers the scenarios where the profit of a job can be correlated with its size and we can cancel a job during its execution in the policy. No profit is gathered from a canceled job. This generalization is referred as *Stochastic Knapsack with Correlated Rewards and Cancelations (SK-CC)*.

Our Results: For SK, Bhalgat, Goel and Khanna [9] obtained a $(1+\epsilon)$-approximation using ϵ extra capacity. We obtain an alternative proof of this result, using the Poisson approximation technique. The running time of our algorithm is $n^{O(f(\epsilon))}$ where $f(\epsilon) = 2^{\text{poly}(1/\epsilon)}$, improving upon the $n^{O(f(\epsilon))^{O(f(\epsilon))}}$ running time in [9]. Our algorithm is also considerably simpler.

THEOREM 1.5. *For any $\epsilon > 0$, there is a polynomial time algorithm that finds a $(1+\epsilon)$-approximate adaptive policy for SK when the capacity is relaxed to $(1+\epsilon)\mathbb{C}$.*

Our next main result is a generalization of Theorem 1.5 to SK-CC where the size and profit of an item may be correlated, and cancelation of item in the middle is allowed. The current best known result for SK-CC is a factor 8 approximation algorithm by Gupta, Krishnaswamy, Molinaro and Ravi [23], base on a new time-indexed LP relaxation. We remark that it is not clear to us how to extend the enumeration technique developed in [9] to handle cancelations.

THEOREM 1.6. *For any $\epsilon > 0$, there is a polynomial time algorithm that finds a $(1+\epsilon)$-approximate adaptive policy for SK-CC when the capacity is relaxed to $(1+\epsilon)\mathbb{C}$.*

We use SK-Can to denote the stochastic knapsack problem where cancelations are allowed (the size and profit of an item are not correlated). Based on Theorem 1.6 and the algorithm in [8], we obtain a generalization of the result in [8] as follows.

THEOREM 1.7. *For any $\epsilon > 0$, there is a poly-time algorithm that finds a $(2+\epsilon)$-approximate adaptive policy for SK-Can.*

Bayesian Online Selection: The technique developed for SK-CC can be used to obtain the following result for an interesting variant of SK, where the size and the profit of an item are revealed before the decision whether to select the item is made. We call this problem the *Bayesian online selection problem subject to a knapsack constraint* (BOSP-KC). The problem falls into the framework of Bayesian online selection problems (BOSP) formulated in [29]. BOSP problems subject to various constraints have attracted a lot of attention due to their applications to mechanism design [24, 12, 1, 29]. BOSP-KC also has a close relation with the *knapsack secretary* problem [3] (See Section 6 for a discussion).

THEOREM 1.8. *For any $\epsilon > 0$, there is a polynomial time algorithm that finds a $(1+\epsilon)$-approximate adaptive policy for BOSP-KC when the capacity is relaxed to $(1+\epsilon)\mathbb{C}$.*

Due to space constraints, we omit many details and proofs, which can be found in the full version of the paper.

1.4 Other Related Work

There is a large body of literature on EUM and TPM, especially for specific combinatorial problems and/or special utility functions. For the EUM version of the stochastic path problem, Nikolova, Brand and Karger [36] identified more specific utility and distribution combinations that can be solved optimally in polynomial time. Nikolova, Kelner, Brand and Mitzenmacher [37] and Nikolova [35] studied the TPM problem when the distributions of the edge lengths are normal, Poisson or exponential. Many heuristics for the stochastic shortest path problems have been proposed to deal with more general utility functions (see e.g., [33, 34, 7]). However, either their running times are exponential in worst cases or there is no provable performance guarantee for the produced solution. The TPM version of the minimum spanning tree problem has been studied in [26, 20], where polynomial time algorithms have been developed for Gaussian distributed edges. Kleinberg, Rabani and Tardos [28] first considered the fixed set version of the stochastic knapsack problem with Bernoulli-type distributions. They provided a polynomial-time logarithimic-approximation. For exponentially distributed items, Goel and Indyk [21] presented a bi-criterion PTAS. For Gaussian distributions, Goyal and Ravi [22] obtained a PTAS.

The adaptive stochastic knapsack problem and several of it variants have been shown to be PSPACE-hard [18], which implies that it is impossible in polynomial time to construct an optimal adaptive policy, which may be exponentially large and arbitrarily complicated. Dean, Goemans, and Vondrak [18] first studied SK from the perspective of approximation algorithms and gave an algorithm with an approximation factor of $3+\epsilon$. In fact, their algorithm produces a non-adaptive policy (a permutation of items) which implies the *adaptivity gap* of the problem, the maximum ratio between the expected values achieved by the best adaptive and non-adaptive strategies, is a constant. Using the technique developed for the $(1+\epsilon)$-approximation using ϵ extra capacity, Bhalgat, Goel and Khanna [9] also gave an improved $(\frac{8}{3}+\epsilon)$-approximation without extra capacity. Stochastic multidimensional knapsack (also called stochastic packing) has been also studied [17, 9, 5].

BOSP problems are often associated with the name *prophet inequalities* since the solutions of the online algorithms are often compared with "the prophet's solutions" (i.e., the offline optimum). The prophet inequalities were proposed in the seminal work of Krengel and Sucheston [30] and have been studied extensively since then. The secretary problem is a also classical online selection problem introduced by Dynkin [19]. Recently, both problems enjoy a revival due to their connections to mechanism design and many generalizations have been studied extensively [4, 3, 25, 11, 27, 24, 12, 1, 29]. We note that performances in all the work mentioned above are measured by comparing the solutions of the online policies with the offline optimum. Complementarily, our work compares our policies with the optimal online policies.

Finally, we would like to point out that Daskalakis and Papadimitrious [16] recently used Poisson approximation in approximating mixed Nash Equilibria in anonymous games. In particular, they showed that a set of Bernoulli random variables can be discretized such that the probability values are integer multiple of $1/k$ for some integer k (k should be a function of ϵ) and the distance between the sum of the discretized variables and that of the original variables is only $O(\sqrt{1/k})$. Therefore, the search space can be reduced to $n^{\text{poly}(k)}$ for their problem. Our technique requires to deal with general distributions (rather than Bernoulli), but allows for poly(n) discretized probability values.

2. EXPECTED UTILITY MAXIMIZATION

We prove Theorem 1.1 and Theorem 1.2 in this section. For each item $b \in \mathbb{B}$, we use $\pi_b(.)$ to denote probability distribution of the weight X_b of b. We use $\mathcal{F} \subseteq 2^{\mathbb{B}}$ to denote the set of feasible solutions. For example, in the minimum spanning tree problem, $\mathbb{B}$ is the set of edges and $\mathcal{F}$ is the set of all spanning trees. Since we are satisfied with an ϵ additive approximation, we can assume w.l.o.g. the utility function $\mu(x) = 0$ if $x \geq \mathbb{C}$ for some constant $\mathbb{C}$ (e.g., we can choose $\mathbb{C}$ to be a constant such that $\mu(x) < \epsilon$ if $x \geq \mathbb{C}$). Thus, the support of $\pi_b(.)$ can be assumed to be a subset of $[0, \mathbb{C}]$. By scaling, we can assume $\mathbb{C} = 1$ and $0 \leq \mu(x) \leq 1$ for $x \geq 0$. We also assume μ is α-Lipschitz where α is a constant that does not depend on ϵ. It is straightforward to extend our analysis to the case where α depends on ϵ. We first consider the general EUM problem and then focus on EUM-Mono where the utility function is monotone nonincreasing.

We start by bounding the total expected size of solution S if $\mu(S)$ is not negligible.

LEMMA 2.1. *Suppose each item $b \in \mathbb{B}$ has a non-negative random weight X_b taking values from $[0, \mathbb{C}]$. Then, $\forall S \subseteq U$, $\forall \frac{1}{2} > \epsilon > 0$, if $\mathbb{E}[\mu(X(S))] \geq \epsilon$, then $\mathbb{E}[X(S)] \leq 3/\epsilon$.*

Let S^* denote the optimal feasible set and OPT the optimal value. If $\mathsf{OPT} = \mathbb{E}[\mu(X(S^*))] \leq \epsilon$, then any feasible solution achieves the desired approximation guarantee since $\mathsf{OPT} - \epsilon \leq 0$. Hence, we focus on the other case where $\mathsf{OPT} > \epsilon$. We call an item b *heavy item* if $\mathbb{E}[X_b] > \epsilon^{10}$. Otherwise we call it *light*. By Lemma 2.1, we can see that the number of heavy items in S^* is at most $\frac{3}{\epsilon^{11}}$.

2.1 Discretization

In this section, we discuss how to discretize the size distributions for items, using parameter ϵ. Our discretization is similar to the one in [9], however, ours is much simpler.

For item b, we say b realizes to a "large" size if $X_b > \epsilon^4$. Otherwise we say b realizes to a "small" size. The discretization consists of two steps. We discretize the small size region in step 1 and the large size region in step 2. We use $\widetilde{X}_b$ to denote the size after discretization and $\widetilde{\pi}_b$ its distribution.

Step 1. Small size region In the small size region, $\widetilde{X}_b$ follows a Bernoulli distribution, taking only values 0 and ϵ^4. The probability values $\Pr[\widetilde{X}_b = 0]$ and $\Pr[\widetilde{X}_b = \epsilon^4]$ are set such that $\mathbb{E}[\widetilde{X}_b \mid X_b \leq \epsilon^4] = \mathbb{E}[X_b \mid X_b \leq \epsilon^4]$.

More formally, suppose w.l.o.g. that there is a value $0 \leq d \leq \epsilon^4$ such that $\Pr[X_b \geq d \mid X_b \leq \epsilon^4] \cdot \epsilon^4 = \mathbb{E}[X_b \mid X_b \leq \epsilon^4]$. We create a mapping between X_b and $\widetilde{X}_b$ as follows:

$$\widetilde{X}_b = \begin{cases} 0, & 0 \leq X_b < d; \\ \epsilon^4, & d \leq X_b \leq \epsilon^4; \end{cases}$$

Step 2. Large size region If X_b realizes to a large size, we simply discretize it as follows: Let $\widetilde{X}_b = \lfloor \frac{X_b}{\epsilon^5} \rfloor \epsilon^5$ (i.e., we round a large size down to a multiple of ϵ^5).

The above two discretization steps are used throughout this paper. We denote the set of the discretized sizes by $\mathcal{S} = \{\mathrm{s}_0, \mathrm{s}_1, \ldots, \mathrm{s}_{z-1}\}$ where $\mathrm{s}_0 = 0, \mathrm{s}_1 = \epsilon^5, \mathrm{s}_2 = 2\epsilon^5, \ldots, \mathrm{s}_{z-1}$. Note that $\mathrm{s}_1 = \epsilon^5, \ldots, \mathrm{s}_{1/\epsilon-1} = \epsilon^4 - \epsilon^5$ are also included in $\mathcal{S}$, even though their probability is 0. It is straightforward to see that $|\mathcal{S}| = z = O(\mathbb{C}/\epsilon^5)$. This finishes the description of the discretization.

The following lemma states that for a set of items, the behavior of the sum of their discretized distributions is very close to that of their original distributions.

LEMMA 2.2. *For any set S of items such that $\mathbb{E}[X(S)] \leq 3/\epsilon$, $\big|\mathbb{E}[\mu(X(S))] - \mathbb{E}[\mu(\widetilde{X}(S))]\big| = O(\epsilon)$.*

2.2 Poisson Approximation

For an item b, we define its *signature* to be the vector

$$\mathsf{Sg}(b) = \big(\overline{\pi}_b(\mathrm{s}_1), \overline{\pi}_b(\mathrm{s}_2), \overline{\pi}_b(\mathrm{s}_3), \ldots, \overline{\pi}_b(\mathrm{s}_{z-1})\big),$$

where $\overline{\pi}_b(\mathrm{s}) = \big\lfloor \widetilde{\pi}_b(\mathrm{s}) \cdot \frac{n}{\epsilon^6} \big\rfloor \cdot \frac{\epsilon^6}{n}$ for all nonzero discretized size $\mathrm{s} \in \mathcal{S} \setminus \{0\} = \{\mathrm{s}_1, \mathrm{s}_2, \ldots, \mathrm{s}_{z-1}\}$. For a set S of items, its *signature* is defined to be the sum of the signatures of all items in S, i.e., $\mathsf{Sg}(S) = \sum_{b \in S} \mathsf{Sg}(b)$. We use $\mathsf{Sg}(S)_k$ to denote the kth coordinate of $\mathsf{Sg}(S)$. By Lemma 2.1, $\sum_{k=1}^{z-1} \mathsf{Sg}(S)_k \cdot \mathrm{s}_k = \sum_{k=1/\epsilon}^{z-1} \mathsf{Sg}(S)_k \cdot \mathrm{s}_k \leq 3/\epsilon$. Thus $\mathsf{Sg}(S)_k \leq 3/\epsilon^5$ for all k. Therefore, the number of possible signatures is bounded by $\big(3n/\epsilon^{11}\big)^{|\mathcal{S}|-1}$.

For an item b, we let $\overline{X}_b$ be the random variable that $\Pr\big[\overline{X}_b = \mathrm{s}\big] = \overline{\pi}_b(\mathrm{s})$ for $\mathrm{s} = \mathrm{s}_1, \mathrm{s}_2, \ldots, \mathrm{s}_{z-1}$, and $\overline{X}_b = 0$ with the rest of the probability mass. Similarly, we use $\overline{X}(S)$ to denote $\sum_{b \in S} \overline{X}_b$ for a set S of items.

The following lemma shows that it is sufficient to enumerate all possible signatures for the set of light items.

LEMMA 2.3. *Let S_1, S_2 be two sets of light items such that $\mathsf{Sg}(S_1) = \mathsf{Sg}(S_2)$ and $\mathbb{E}[\widetilde{X}(S_1)] \leq 3/\epsilon, \mathbb{E}[\widetilde{X}(S_2)] \leq 3/\epsilon$. Then, the total variation distance between $X(S_1)$ and $X(S_2)$ $\Delta\big(\widetilde{X}(S_1), \widetilde{X}(S_2)\big) = O(\epsilon)$.*

The following Poisson approximation theorem by Le Cam [31], rephrased in our language, is essential for proving Lemma 2.3. Suppose we are given a K-dimensional vector $V = (V_1, \ldots, V_K)$. Let $\lambda = \sum_{i=1}^{K} V_i$. we say a random variable Y follows the *compound Poisson distribution corresponding to V* if it is distributed as $Y = \sum_{j=1}^{N} Y_j$ where N follows Poisson distribution with expected value λ (denoted as $N \sim \text{Pois}(\lambda)$) and $Y_1, \ldots, Y_N$ are i.i.d. random variables with $\Pr[Y_j = 0] = 0$ and $\Pr[Y_j = k] = V_k/\lambda$ for $k \in \{1, \ldots, K\}$ and $j \in \{1, \ldots, N\}$.

LEMMA 2.4. [31] *Let $X_1, X_2, \ldots$ be independent random variables taking integer values in $\{0, 1, ..., K\}$, let $X = \sum X_i$. Let $\pi_i = \Pr[X_i \neq 0]$ and $V = (V_1, \ldots, V_K)$ where $V_k = \sum_i \Pr[X_i = k]$. Suppose $\lambda = \sum_i \pi_i = \sum_k V_k < \infty$. Let Y be the compound Poisson distribution corresponding to vector V. Then, the total variation distance between X and Y can be bounded as follows:*

$$\Delta\Big(X, Y\Big) = \frac{1}{2} \sum_{k \geq 0} \Big|\Pr[X = k] - \Pr[Y = k]\Big| \leq 2 \sum_i \pi_i^2.$$

2.3 Approximation Algorithm of EUM

Now, everything is in place to present our approximation algorithm. We first discretize the size distributions of for all light items. Then, we enumerate all possible heavy item sets H with $\mathbb{E}\big[\widetilde{X}(H)\big] < 3/\epsilon$. For each such set H, we do the following. We enumerate all possible signatures Sg. For each Sg, we try to find a set L of light items such that $H \cup L \in \mathcal{F}$ ($H \cup L$ is feasible) and $\mathsf{Sg}(L) = \mathsf{Sg}$. Here we use the pseudopolynomial time algorithm for the exact version of the problem. Finally, we output the feasible $H \cup L$ with the largest $\mathbb{E}\big[\mu\big(\widetilde{X}(H \cup L)\big)\big]$

Since the number of different H is bounded by $n^{O(\epsilon^{-11})}$ and the number of different signatures is at most $n^{O(\epsilon^{-5})}$, the algorithm clearly runs in $n^{\mathrm{poly}(1/\epsilon)}$ time overall. Finally, we present the analysis of the performance guarantee.

Proof of Theorem 1.1: Assume the optimal feasible set is $S^* = H^* \cup L^*$ where items in H^* are heavy and items in L^* are light. Assume our algorithm has guessed H^* correctly. Since there is a pseudopolynomial algorithm, we can find a set L of light items such that $\mathsf{Sg}(L) = \mathsf{Sg}(L^*)$. By Lemma 2.3, we know that $\Delta\Big(\widetilde{X}(L), \widetilde{X}(L^*)\Big) = O(\epsilon)$. Therefore, we can get that $\Delta\Big(\widetilde{X}(L \cup H^*), \widetilde{X}(L^* \cup H^*)\Big) = O(\epsilon)$. Moreover, we have that

$$\begin{aligned}
&\Big|\,\mathbb{E}\Big[\mu\big(\widetilde{X}(L \cup H^*)\big)\Big] - \mathbb{E}\Big[\mu\big(\widetilde{X}(L^* \cup H^*)\big)\Big]\,\Big| \\
\le &\sum_{\mathrm{s} \in \mathcal{S}} \mu(\mathrm{s}) \left|\widetilde{\pi}_{L \cup H^*}(\mathrm{s}) - \widetilde{\pi}_{L^* \cup H^*}(\mathrm{s})\right| \\
\le &\max_{0 \le x \le \mathbb{C}} \mu(x) \cdot \Delta\Big(\widetilde{X}(L \cup H^*), \widetilde{X}(L^* \cup H^*)\Big) = O(\epsilon),
\end{aligned}$$

where $\widetilde{\pi}_S$ is the PDF for $\widetilde{X}(S)$. Now, we have

$$\begin{aligned}
&|\,\mathbb{E}[\mu(X(L^* \cup H^*))] - \mathbb{E}[\mu(X(L \cup H^*))]\,| \\
\le &\Big|\,\mathbb{E}[\mu(X(L^* \cup H^*))] - \mathbb{E}[\mu(\widetilde{X}(L^* \cup H^*))]\,\Big| \\
&+ \Big|\,[\mathbb{E}[\mu(\widetilde{X}(L^* \cup H^*)) - \mathbb{E}[\mu(\widetilde{X}(L \cup H^*))]]\,\Big| \\
&+ \Big|\,\mathbb{E}[\mu(\widetilde{X}(L \cup H^*)) - \mathbb{E}[\mu(X(L \cup H^*))]]\,\Big| = O(\epsilon)
\end{aligned}$$

The inequality follows from Lemma 2.2 and 2.3. □

2.4 Approximation Algorithm for EUM-Mono

We prove Theorem 1.2 in this subsection. Recall that EUM-Mono is a special case of EUM where the utility function μ is monotone nonincreasing. The algorithm is the same as that in EUM except that for each Sg, we try to find a set L of light items such that $H \cup L \in \mathcal{F}$ and $\mathsf{Sg}(L) \le (1+\epsilon^6)\mathsf{Sg}$ (coordinatewise) using the multidimensional PTAS.

LEMMA 2.5. *Let S_1, S_2 be two sets of light items with $\mathbb{E}\big[\widetilde{X}(S_1)\big] \le 3/\epsilon$ and $\mathbb{E}\big[\widetilde{X}(S_2)\big] \le 3/\epsilon$. If $\mathsf{Sg}(S_1) \le (1+\epsilon^6)\mathsf{Sg}(S_2)$, then we have that for any $\beta > 0$*

$$\Pr[\widetilde{X}(S_1) \le \beta] \ge \Pr[\widetilde{X}(S_2) \le \beta] - O(\epsilon).$$

PROOF. (sketch) Let Z_1 and Z_2 be the compound Poisson distribution (CPD) corresponding to $\mathsf{Sg}(S_1)$ and $\mathsf{Sg}(S_2)$, respectively. Denote $\lambda = \|\mathsf{Sg}(S_2)\|_1$. Let Y be the CPD defined as $Y = \sum_{i=1}^{N} Y_i$ where $N \sim \mathrm{Pois}((1+\epsilon^6)\lambda)$ and Y_is are i.i.d. random variables with $\Pr[Y_i = \mathrm{s}_k] = \frac{\mathsf{Sg}(S_2)_k}{\lambda}$ for each $\mathrm{s}_k \in \mathcal{S} \setminus \{\mathrm{s}_0\}$. By Lemma 2.4, Z_2 is distributed as $\sum_{i=1}^{N'} Y_i$ where $N' \sim \mathrm{Pois}(\lambda)$. By the standard coupling argument, we can see that

$$\Delta(Y, Z_2) \le \Delta(N, N') = \Delta(\mathrm{Pois}((1+\epsilon^6)\lambda), \mathrm{Pois}(\lambda)) = O(\epsilon).$$

Since $\mathsf{Sg}(S_1) \le (1+\epsilon^6)\mathsf{Sg}(S_2)$, Z_1 (the CPD corresponding to $\mathsf{Sg}(S_1)$) is stochastically dominated by Y (the CPD corresponding to $(1+\epsilon^6)\mathsf{Sg}(S_2)$). Therefore,

$$\begin{aligned}
\Pr[\overline{X}(S_1) \le \beta] &\ge \Pr[Z_1 \le \beta] - \Delta(\overline{X}(S_1), Z_1) \\
&\ge \Pr[Y \le \beta] - O(\epsilon) \\
&\ge \Pr[Z_2 \le \beta] - \Delta(Y, Z_2) - O(\epsilon) \\
&\ge \Pr[\overline{X}(S_2) \le \beta] - \Delta(\overline{X}(S_2), Z_2) - O(\epsilon) \\
&\ge \Pr[\overline{X}(S_2) \le \beta] - O(\epsilon)
\end{aligned}$$

We also have $\Delta\Big(\widetilde{X}(S), \overline{X}(S)\Big) \le \epsilon$ for $S = S_1, S_2$. Thus

$$\Pr[\widetilde{X}(S_1) \le \beta] \ge \Pr[\widetilde{X}(S_2) \le \beta] - O(\epsilon).$$

This completes the proof of the lemma. □

The proof of Theorem 1.2 is similar to that of Theorem 1.1 except that we use Lemma 2.5 instead of Lemma 2.3.

3. STOCHASTIC BIN PACKING

Recall that in the stochastic bin packing (SBP) problem, we are given a set of items $\mathbb{B} = \{b_1, b_2, \ldots, b_n\}$ and an overflow probability $0 < p < 1$. The size of each item b_i is an independent random variable X_i. The goal is to pack all the items in $\mathbb{B}$ into bins with capacity $\mathbb{C}$, using as few bins as possible, such that the overflow probability for each bin is at most p. The main goal of this section is to prove Theorem 1.3.

W.l.o.g., we can assume that $p \le 1 - \epsilon$ where ϵ is the error parameter. Otherwise, the overflow probability is relaxed to $p + \epsilon \ge 1$, and we can pack all items in a single bin. Let the number of bins used in the optimal solution be m. In our algorithms, we relax the bin size to $\mathbb{C} + O(\epsilon)$, which is less than $2\mathbb{C}$. W.l.o.g., we assume the support of X_i is $[0, 2\mathbb{C}]$. Otherwise, we can truncate the distribution. From now on, assume that our algorithm has guessed m correctly. We use $B_1, B_2, \ldots, B_m$ to denote the bins.

3.1 Discretization

We first discretize the size distributions for all items in $\mathbb{B}$, using parameter ϵ, as described in Section 2.1. Denote the discretized size of b_i by $\widetilde{X}_i$.

We call item b_i a *heavy* item if $\mathbb{E}[X_i] \ge \epsilon^{15}$. Otherwise, b_i is *light*. We need to further discretize the size distributions of the heavy items. We round down the probabilities of $\widetilde{X}_i$ taking each nonzero value to multiples of ϵ^{22}. Denote the resulting random size by $\widehat{X}_i$. More formally, $\Pr[\widehat{X}_i = \mathrm{s}] = \lfloor \Pr[\widetilde{X}_i = \mathrm{s}] \cdot \epsilon^{-22} \rfloor \cdot \epsilon^{22}$ for any $\mathrm{s} \in \mathcal{S} \setminus \{0\}$. Use $\mathcal{H}$ to denote the set of all discretized distributions for heavy items. We can see that $|\mathcal{H}| = (\epsilon^{-22})^{|\mathcal{S}|} = (\epsilon^{-22})^{O(\epsilon^{-5})}$. Denote them by $\Pi_1, \Pi_2, \ldots, \Pi_{|\mathcal{H}|}$ (in an arbitrary order).

For a set S of items, we use $H(S)$ to denote the set of heavy items in S, and use $L(S)$ to denote the set of light items in S. We define the *arrangement* for heavy items in S to be the $|\mathcal{H}|$-dimensional vector: $\mathsf{Ar}(S) = (N_1, N_2, \ldots, N_{|\mathcal{H}|})$ where $N_k \in \mathbb{N}$ is the number of heavy items in S following the discretized size distribution Π_k, $k = 1, 2, \ldots, |\mathcal{H}|$. Suppose we pack all items in S into one bin. By Lemma 2.1 and the assumption that $p \le 1 - \epsilon$, $\mathbb{E}\big[\widetilde{X}(S)\big] \le 3/\epsilon$. So, we

can pack at most $O(\epsilon^{-16})$ heavy items into a bin. Therefore, the number of possible arrangements for a bin is bounded by $\binom{|\mathcal{H}|+O(\epsilon^{-16})}{|\mathcal{H}|}$, which is a constant.

Let the *signature* of a light item b be $\mathsf{Sg}(b) = \big(\widetilde{\pi}_b(\mathrm{s}_i)\big)_{1\le i\le|\mathcal{S}|-1}$, (Note that the definition is slightly different from the previous one). The *signature* of the a set S of light items is defined to be $\mathsf{Sg}(S) = \sum_{b\in S}\mathsf{Sg}(b)$. If S consists of both heavy and light items, we use $\widehat{\mathsf{Sg}}(S)$ as a short for $\mathsf{Sg}(L(S))$. Moreover, for set S, we define the *rounded signature* to be

$$\widehat{\mathsf{Sg}}(S) : \widehat{\mathsf{Sg}}(S)_k = \big\lceil \mathsf{Sg}(S)_k \cdot \epsilon^{-6}\big\rceil\cdot\epsilon^6 \quad \text{for } k = 1,\dots,|\mathcal{S}|-1.$$

Suppose we pack all items in S into one bin. Since $\mathbb{E}\big[\widetilde{X}(S)\big] \le 3/\epsilon$, $\widehat{\mathsf{Sg}}(S)_k \le O(1/\epsilon^5)$ for any k. Therefore, the number of possible rounded signatures is bounded by $(3/\epsilon^{11})^{|\mathcal{S}|-1}$.

The *configuration* of a set S of items is defined to be $\mathsf{Cf}(S) = \big(\mathsf{Ar}(S); \widehat{\mathsf{Sg}}(S)\big)$. We can see the number of all configurations is bounded by $h = O\big(\frac{1}{\epsilon^{11}}\big)^{|\mathcal{S}|-1}\binom{|\mathcal{H}|+O(\epsilon^{-16})}{|\mathcal{H}|}$.

We also define the *s-configuration* of a solution $\mathsf{SOL} = \{S_i\}_{i=1,\dots,m}$ (S_m is the set of items packed in bin B_i) to be

$$\mathsf{Cf}(\mathsf{SOL}) = \big\{\mathsf{Cf}(S_1), \mathsf{Cf}(S_2), \dots, \mathsf{Cf}(S_m)\big\}.$$

We note that $\mathsf{Cf}(\mathsf{SOL})$ is a multi-set (instead of a vector), i.e., the indices of the bins do not matter. Hence, the number of all possible s-configurations is bounded by $\binom{m+h}{m} = \mathrm{poly}(m)$. Let $\mathcal{C}$ be the set of all possible s-configurations.

3.2 Our Algorithm

Before describing our algorithm, we need a procedure to solve the following multi-dimensional optimization problem: Given an s-configuration $\mathsf{Cf} = \big((\mathsf{Ar}_1, \widehat{\mathsf{Sg}_1}), \dots, (\mathsf{Ar}_m, \widehat{\mathsf{Sg}_m})\big)$, find a packing $\mathsf{SOL} = (S_1, \dots, S_m)$ such that $\mathsf{Ar}(S_i) = \mathsf{Ar}_i$ and $\mathsf{Sg}(S_i) \le \widehat{\mathsf{Sg}_i} + \epsilon^6\mathbf{1}$ (where $\mathbf{1} = (1,\dots,1)$) for $1 \le i \le m$ or to claim that there is no packing $\mathsf{SOL} = (S_1,\dots,S_m)$ such that $\mathsf{Ar}(S_i) = \mathsf{Ar}_i$ and $\mathsf{Sg}(S_i) \le \widehat{\mathsf{Sg}_i}$ for $1 \le i \le m$. If we succeed in finding such a solution, we say Cf *passes the test of feasibility*, otherwise we say Cf fails.

Finding a solution such that $\mathsf{Ar}(S_i) = \mathsf{Cf}_i$ for all i is trivial. Now, we concentrate on the set of the light items, $L(\mathbb{B})$. In fact, the problem for light items becomes a variant of the multidimensional bin packing problem, called the *vector scheduling* problem, which has been studied in [13]. For completeness, we sketch their approach, using our notations. We write a linear integer program, solve its LP relaxation and then round the solution to a feasible packing. We use the Boolean variables x_{ij} to denote whether the light item b_i is packed into bin B_j. We have the integrality constraints $x_{ij} \in \{0,1\}$ and they are relaxed to $x_{ij} \ge 0$ in the following LP relaxation:

1. $\sum_{j=1}^{m} x_{i,j} = 1, \qquad i \in L(\mathbb{B}),$
2. $\sum_{i\in L(\mathbb{B})} \widetilde{\pi}_i(\mathrm{s}_k)\cdot x_{ij} \le (\widehat{\mathsf{Sg}_j})_k, \qquad j \in \{1,\dots,m\},\ k \in \{1,\dots,|\mathcal{S}|-1\},$
3. $x_{ij} \ge 0, \qquad i \in L(\mathbb{B}),\ j \in \{1,\dots,m\}.$

If the above LP has no feasible solutions, we say Cf fails the test. Otherwise Cf passes the test, and we find a solution SOL as follows. First, we solve the LP and obtain a basic feasible solution. It is well known that any *basic feasible solution* to the LP has at most $(|\mathcal{S}|-1)\cdot m$ light items that are packed fractionally into more than one bins. Let F be the set of light items that are fractionally packed into more than bins. $|F| \le (|\mathcal{S}|-1)\cdot m$. We partition F arbitrarily into m subsets, each containing at most $(|\mathcal{S}|-1)$ items, and then pack the k-th subset into the k-th bin. Since the expected size of a light item is less than ϵ^{15}, $\widetilde{\pi}_i(\mathrm{s}_k) \le \epsilon^{11}$ for any i,k. Therefore, $\mathsf{Sg}(S_j) \le \widehat{\mathsf{Sg}}_j + (|\mathcal{S}|-1)\epsilon^{11} = \widehat{\mathsf{Sg}}_j + \epsilon^6\mathbf{1}, \forall j$.

We need one more notation to describe our algorithm. For a set of items S, let

$$\Pr(\mathsf{Cf}(S),\mathbb{C}) = \Pr((\mathsf{Ar}(S),\widehat{\mathsf{Sg}}(S)),\mathbb{C}) = \Pr\big[\widehat{X}_H + \widehat{Y}_L \ge \mathbb{C}\big]$$

where $\widehat{X}_H = \widehat{X}\big(H(S)\big)$ and $\widehat{Y}_L$ is the CPD corresponding to $\widehat{\mathsf{Sg}}(S)$. By definition, if two sets S_1 and S_2 have the same configuration, $\Pr(\mathsf{Cf}(S_1),\mathbb{C}) = \Pr(\mathsf{Cf}(S_2),\mathbb{C})$.

Now, everything is ready to state our algorithm. We simply enumerate all s-configurations in $\mathcal{C}$. For each s-configuration $\mathsf{Cf} = \{\mathsf{Cf}_1,\dots,\mathsf{Cf}_m\}$, we first compute $\Pr(\mathsf{Cf}_i,\mathbb{C})$ for all $i = 1,\dots,m$. If all of them are at most $p + O(\epsilon)$, we run the feasibility test. If Cf passes the test, the returned solution is our final packing.

The algorithm clearly runs in polynomial time since the number of s-configurations is polynomial and the feasibility test also runs in polynomial time. The analysis of the performance guarantee appears in the full version.

Without Relaxing the Overflow Probability: For SBP, we can find a 3-approximation within polynomial time when we relax the bin size by ϵ but do not relax the overflow probability p, for any constant $0 < p < 1$. This improves the $O(1/\epsilon)$-approximation in [28]. This is done by first pack the items into $B_1,\dots,B_m$ such that $\Pr[X(B_i) \ge 1+\epsilon] \le p+\epsilon$. Then we distribute the items in B_i to at most 3 new bins: We pack each item in B_i into a new bin $B_{i,j}$ While there exist two bins B_{i,j_1}, B_{i,j_2} such that the estimate value $\Pr[X(B_{i,j_1}\cup B_{i,j_2}) \le \mathbb{C}+\epsilon]$ is no more than $(1-\epsilon)p$, we merge B_{i,j_1} and B_{i,j_2} into one bin. We can show that for each B_i, the process terminates with at most 3 new bins.

4. STOCHASTIC KNAPSACK

An instance of the stochastic knapsack problem can be specified by a tuple $(\pi,\mathbb{C})$, where $\pi = \{\pi_1,\pi_2,...,\pi_n\}$. π_i is the joint distribution of size and profit for item b_i, $\forall i$. $\mathbb{C}$ is the capacity of the knapsack. W.l.o.g., we can assume $\frac{1}{2} \le \mathbb{C} \le 2$ and the size of each item is distributed between 0 and $(1+\epsilon)\mathbb{C}$. The distributions for different items are mutually independent. We let random variables X_b and P_b denote the size and profit of item b. We use π_b to denote the probability distribution of X_b, i.e., $\pi_b(w) = \Pr[X_b = w]$. W.l.o.g., we can assume that for each item, we obtain a fixed profit for each realized size. For item b, we define the *effective profit* function p_b to be $p_b(w) = \mathbb{E}[P_b \mid X_b = w]\cdot \Pr[X_b = w]$ for any $w \in [0,\mathbb{C}]$. We let $p_b(I) = \sum_{w\in I} p_b(w)$ for any $I \subseteq [0,\mathbb{C}]$.

Policies The process of applying a policy σ on an instance $(\pi,\mathbb{C})$ can be represented as a decision tree $T(\sigma,\pi,\mathbb{C})$. Each node v in T_σ corresponds to placing an item in the knapsack. Each edge $e = (v,u)$ in T_σ (v is the parent) corresponds to a size realization of v. We use w_e, π_e to denote the corresponding size and probability of e, respectively. We also use T_σ to denote $T(\sigma,\pi,\mathbb{C})$ when the context is clear.

We call the path from root to v in T_σ the *realization path* of v, and denote it by $R(v)$. We use $\mathbb{P}(\sigma,\pi,\mathbb{C})$ to denote the expected profit that the policy σ can obtain with the given distributions π and total capacity $\mathbb{C}$. We also use the

shorthand notation $\mathbb{P}(\sigma)$ or $\mathbb{P}(T_\sigma)$ if the context is clear. The expected profit of the subtree T_v rooted at v can be computed recursively as

$$\mathbb{P}(v) = \sum_{e=(v,u)|W(v)+w_e\le\mathbb{C}} \big[p_v(w_e) + \pi_e \cdot \mathbb{P}(u)\big].$$

The expected profit $\mathbb{P}(\sigma)$ of policy σ is simply $\mathbb{P}$(the root of T_σ). We use OPT to denote the expected profit of the optimal adaptive policy. We note that in some steps of our algorithm, we assume knowledge of OPT. In fact, any constant approximation of OPT, which can be obtained using the approximation algorithm in [18] for SK or the one in [23] for $\mathsf{SK\text{-}CC}$, would suffice for our purpose.

We start with some simple properties shown in Bhalgat et al. [9] [1]. W.l.o.g., we also assume that all (optimal or near optimal) policies σ considered in this paper have the following property:

P1. For $u, v \in T_\sigma$, if u is an ancestor of v, then $\mathbb{P}(u) \ge \mathbb{P}(v)$.

Otherwise, replacing the subtree T_u with T_v increases the profit of the policy σ.

LEMMA 4.1 (PART OF LEMMA 2.4 IN [9]). *For any policy σ on instance $(\pi, \mathbb{C})$, there exists a policy σ' such that $\mathbb{P}(\sigma', \pi, \mathbb{C}) = (1 - O(\epsilon))\mathbb{P}(\sigma, \pi, \mathbb{C})$ and*

P2. *for any realization path R, $\sum_{v\in R} \mathbb{E}[X_v] = O(\mathbb{C}/\epsilon)$.*

4.1 Discretization

In this section, we discuss how to discretize the size and profit distributions for items in $\mathbb{B}$, using parameter ϵ. W.l.o.g., we assume that the range of X_b is $[0, \mathbb{C} + \epsilon)$ for any item b. The discretization of the size distributions is the same as the one in Section 2.1. We also need to discretize the profit distributions. For each item b and $w \in [0, \mathbb{C} + \epsilon)$, we use $D_b(w)$ to denote the mapping from w to its discretized version, i.e., $D_b(w)$ is the value of $\widetilde{X}_b$ when $X_b = w$. The discretized effective profit function $\widetilde{p}_b$ is defined to be $\widetilde{p}_b(\mathrm{s}) = \sum_{w|D_b(w)=\mathrm{s}} p_b(w)$, for all $\mathrm{s} \in \mathcal{S}$. This finishes the description of the discretization step

We need the notion of *canonical policies* introduced in [9]. A policy $\widetilde{\sigma}$ is a canonical policy if it makes decisions based on the discretized sizes of items inserted, rather than their actual sizes. A canonical policy stops inserting items when the total discretized size of items inserted exceeds the knapsack capacity $\mathbb{C}$. Before that, it attempts to insert items even if the total actual size overflows. No profit from those items can be collected. In this following lemma, we show it suffices to only consider canonical policies. The proof is similar to that of Lemma A.5 in [9]. Due to the presence of the correlations between profits and sizes, we need to be more careful in bounding the profit loss.

LEMMA 4.2. *Let π be the joint distribution of size and profit for items in $\mathbb{B}$ and $\widetilde{\pi}$ be the discretized version of π. Then, the following statements hold:*

1. *For any policy σ, there exists a canonical policy $\widetilde{\sigma}$ such that $\mathbb{P}(\widetilde{\sigma}, \widetilde{\pi}, (1 + 4\epsilon)\mathbb{C}) = (1 - O(\epsilon))\mathbb{P}(\sigma, \pi, \mathbb{C})$;*
2. *For any canonical policy $\widetilde{\sigma}$, $\mathbb{P}(\widetilde{\sigma}, \pi, (1 + 4\epsilon)\mathbb{C}) = (1 - O(\epsilon))\mathbb{P}(\widetilde{\sigma}, \widetilde{\pi}, \mathbb{C})$.*

[1] In fact, they only considered the basic version of stochastic knapsack where the profit is a fixed value for an item. However, a scrutiny of their proofs shows that correlations do not affect the properties.

4.2 Block-Adaptive Policies

To further reduce the search space, Bhalgat et al. [9] discovered a very specific class of canonical policies, called *block-adaptive policies* and showed it is sufficient to restrict the search to this set if we are satisfied with a nearly optimal policy. In a block-adaptive policy, instead of inserting one item at a time, we insert a set of items together each time. This set of items is called a *block*. A block-adaptive policy $\widehat{\sigma}$ can also be thought as a decision tree $T_{\widehat{\sigma}}$ where each node in the tree corresponds to a block. Each edge incident on a vertex corresponds to a realization of the sum of the discretized sizes of all items in the block. The following lemma, shown in [9] for SK, summarizes the properties we need.

LEMMA 4.3. *A canonical policy $\widetilde{\sigma}$ with expected profit $(1-\epsilon)\mathsf{OPT}$ can be transformed into a block-adaptive policy with expected profit $(1 - O(\epsilon))\mathsf{OPT}$ when the capacity is further relaxed by $\epsilon\mathbb{C}$. Moreover, the block-adaptive policy satisfies the following properties:*

B1. There are $O(\epsilon^{-14})$ blocks on any root-leaf path in the decision tree.

B2. There are $|\mathcal{S}| = O(\mathbb{C}/\epsilon^5)$ children for each block.

B3. Each block M with more than one items satisfies that $\sum_{b\in M} \mathbb{E}\big[\widetilde{X}_b\big] \le \epsilon^{13}$. [2]

4.3 Poisson Approximation

To search for the (nearly) optimal block-adaptive policy, we want to enumerate all possible structures for a block. In [9], this is done by enumerating all different combinations of the profit contributions from $O(\log n)$ equivalence classes of items, using the technique developed in [14]. Instead, we enumerate all possible signatures of a block, similar to what we have done in Section 2. Since we consider correlated profits and sizes, a signature needs to reflect the profit distribution as well as the size distribution. Formally, for an item b, we define the *signature* of b to be

$$\mathsf{Sg}(b) = \Big(\overline{p}_b(\mathrm{s}_0), \ldots, \overline{p}_b(\mathrm{s}_{z-1}); \overline{\pi}_b(\mathrm{s}_1), \ldots, \overline{\pi}_b(\mathrm{s}_{z-1})\Big),$$

where $\overline{p}_b(\mathrm{s}) = \big\lfloor \widetilde{p}_b(\mathrm{s}) \cdot \frac{n}{\epsilon^{23}\mathsf{OPT}} \big\rfloor \cdot \frac{\epsilon^{23}\mathsf{OPT}}{n}$ and $\overline{\pi}_b(\mathrm{s}) = \big\lfloor \widetilde{\pi}_b(\mathrm{s}) \cdot \frac{n}{\epsilon^{23}} \big\rfloor \cdot \frac{\epsilon^{23}}{n}$ for any $\mathrm{s} \in \mathcal{S}$. For a block M of items, we define the *signature* of M to be $\mathsf{Sg}(M) = \sum_{b\in M} \mathsf{Sg}(b)$. The following lemma is the most crucial to us, which shows that it is sufficient to enumerate all signatures for all blocks in a block-adaptive policy.

LEMMA 4.4. *Consider two decision trees T_1, T_2 corresponding to block-adaptive policies with the same topology (i.e., T_1 and T_2 are isomorphic). If for each block M_1 in T_1, the block M_2 at the corresponding position in T_2 satisfies that $\mathsf{Sg}(M_1) = \mathsf{Sg}(M_2)$, then $|\mathbb{P}(T_1) - \mathbb{P}(T_2)| = O(\epsilon)\mathsf{OPT}$.*

The number of signatures for a block is $\big(O(n/\epsilon^{23})\big)^{O(\epsilon^{-5})} = n^{\mathrm{poly}(1/\epsilon)}$. For any block adaptive policy σ^b, there are at most $f(\epsilon) = \big(O(\epsilon^{-5})\big)^{O(\epsilon^{-14})} = 2^{\mathrm{poly}(1/\epsilon)}$ blocks in its decision tree, since the height of the tree is $O(\epsilon^{-14})$ and the branching factor is at most $|\mathcal{S}| = O(\epsilon^{-5})$. Therefore, the number of all topologies of the decision tree is a constant.

[2] In fact, this property was not explicitly mentioned in Bhalget et al. [9]. But it can be concluded from their construction.

4.4 Enumeration

We have shown it suffices to enumerate over all topologies of the decision trees along with all possible signatures for each block (the number of all possibilities is $n^{O(f(\epsilon))} = n^{2^{\mathrm{poly}(1/\epsilon)}}$) in order to find a nearly optimal block-adaptive policy. Now, we show how to find a nearly optimal block-adaptive policy with a given tree topology along with the signatures for all blocks, using dynamic programming.

The dynamic program is fairly standard and we present a sketch here. Assume the tree topology has been fixed. A configuration C in the dynamic program is a set of signatures, each corresponding to a block in the tree. As we have shown, the number of configurations is poly(n). We use $\mathrm{DP}(i, \mathsf{C}) = 1$ to denote the fact that we can reach configuration C using a subset of $\{b_1, \ldots, b_i\}$. Otherwise, $\mathrm{DP}(i, \mathsf{C}) = 0$. Initially, $\mathrm{DP}(0, \mathbf{0}) = 1$. We compute all $\mathrm{DP}(i, \mathsf{C})$ values in an lexicographically increasing order of (i, C). The value of $\mathrm{DP}(i, \mathsf{C})$ can be computed from the values of $\mathrm{DP}(i-1, \mathsf{C}')$ for all $\mathsf{C}' \le \mathsf{C}$ (coordinatewise). In fact, this step can be done as follows. Suppose we want to compute $\mathrm{DP}(i, \mathsf{C})$. We can decide to place item b_i in a few blocks in the decision tree. The constraint here is no two blocks where we place b_i have an ancestor-descendant relationship. Since the size of tree is $f(\epsilon) = (1/\epsilon^5)^{O(1/\epsilon^{15})}$, so the number of possible ways of adding item b_i is $2^{f(\epsilon)}$ which is (still) a constant. For a particular placement of b_i, we subtract the contribution of b_i from configuration C (i.e., subtract $\mathsf{Sg}(b_i)$ from the vectors in C corresponding to the blocks where we place b_i), resulting another configuration C'. We let $\mathrm{DP}(i, \mathsf{C}) \leftarrow \max(\mathrm{DP}(i, \mathsf{C}), \mathrm{DP}(i-1, \mathsf{C}'))$. Since the number of tree topologies is a constant, the number of configurations is $n^{f(\epsilon)}$ and computing each $\mathrm{DP}(i, \mathsf{C})$ values takes a constant time, the overall running of our algorithm is $O(n^{f(\epsilon)})$, which improves upon the $n^{O(f(\epsilon))^{O(f(\epsilon))}}$ running time in [9].

Combining both results in Lemma 4.2 and 4.3, we can show the main result of this section, Theorem 1.5.

5. HANDLING CANCELATIONS

Recall in the stochastic knapsack problem with correlations and cancelations (SK-CC), we can cancel a job in the middle and we gain zero profit from a canceled job. If we decide to cancel job b after running for t time units, job b can be thought as a job b^t with size $X_{b^t} = \min\{X_b, t\}$, where X_b is the size of b. The effective profit of the new job $p_{b^t}(x)$ equals $p_b(x)$ if $x < t$ and 0 if $x \ge t$. Since we consider discrete time distributions, it only makes sense to cancel a job after a discrete point with nonzero probability mass. Suppose the size of the support of each time distribution is bounded by m. Therefore, for each job b, we can use a set of m jobs to represent all possible cancelations of b, and in each realization path, we are allowed to choose at most one job from the set. In fact, we solve the following more general problem. We have n sets of items, $\mathbb{B}_i, i = 1, 2, ..., n$. Each set $\mathbb{B}_i$ consists of several items $b_{i1}, b_{i2}, ...$ Our goal is to find a policy that packs at most one item from each item set $\mathbb{B}_i$ to the knapsack with capacity $\mathbb{C}$ such that the expected profit is maximized. We call this problem the *generalized stochastic knapsack* problem, denoted as GenSK.

It is not clear to us how to use the technique in Bhalgat et al. [9] to handle the above problem, due to the mutual exclusive relations. However, our technique can be easily extended to GenSK.

5.1 Block-Adaptive Policies

In fact, we can show that Lemma 4.3 also holds for GenSK (thus also for SK-CC). We note that the proof in [9] does not generalize to GenSK and we need to modify the proof in some essential way. We note that our proof works even when there are arbitrary precedence or cardinality constraints imposed on the items. In fact, any realization path of the block-adaptive policy we construct corresponds to some realization path of the original policy $\widetilde{\sigma}$. The ideas may be useful in showing the existence of nearly optimal block-adaptive policies for other problems, thus may be of independent interest.

PROOF. (sketch) Consider the decision tree $T_{\widetilde{\sigma}}$. For any node v and size $\mathrm{s} \in \mathcal{S}$, we use v_{s} to denote the s-*child* of v, that is the child of v corresponding to the size realization s. We define the *segment* starting with node v (denoted as $\mathsf{seg}(v)$) in $T_{\widetilde{\sigma}}$ as the maximal prefix of the leftmost path starting from v such that:

1. If $\mathbb{E}[\widetilde{X}(v)] > \epsilon^{13}$, $\mathsf{seg}(v)$ is the singleton node v. Otherwise, $\mathbb{E}[\widetilde{X}(\mathsf{seg}(v))] \le \epsilon^{13}$;
2. For any two nodes $u, w \in \mathsf{seg}(v)$, and any size s,
$$|\mathbb{P}(T_{u_{\mathrm{s}}}) - \mathbb{P}(T_{w_{\mathrm{s}}})| \le \epsilon^5 \mathsf{OPT}.$$

We greedily partition $T_{\widetilde{\sigma}}$ into segments. Using P2 and the fact that a subtree starting with a smaller capacity and less items has no more profit, we can show B1.

We are ready to describe the algorithm, which takes a canonical policy $\widetilde{\sigma}$ as input, and packs items in a block-adaptive way. For any segment $\mathsf{seg}(v)$, we use $l(v)$ to denote the last node in $\mathsf{seg}(v)$. Similar to the argument in [9], we use two knapsacks, the main knapsack with capacity $\mathbb{C}$ and the auxiliary knapsack with capacity $\epsilon\mathbb{C}$. S represents the auxiliary knapsack.

We start at the root of $T_{\widetilde{\sigma}}$.
repeat
1 Suppose we are at node v in $T_{\widetilde{\sigma}}$. Add the items in $\mathsf{seg}(v)$ to the main knapsack one by one until some node u realizes to a nonzero size, say s.
2 Add all remaining items in $\mathsf{seg}(v)$ to S.
3 Visit node $l(v)_{\mathrm{s}}$.
4 If all nodes in $\mathsf{seg}(v)$ realize to size 0, visit $l(v)_0$.
until A leaf in $T_{\widetilde{\sigma}}$ is reached.
If S overflows, discard the entire profit.

We can see that the set of items the algorithm attempts to insert always corresponds to some realization path in the original policy $\widetilde{\sigma}$. Now we show that the expected profit that new policy can obtain is at least $(1-O(\epsilon))\mathsf{OPT}$. Let us focus on the profit we collect from the main knapsack and ignore those from the auxiliary knapsack. We note that the main knapsack never overflows. Our algorithm deviates the policy $\widetilde{\sigma}$ whenever some node u in the middle of some segment $\mathsf{seg}(v)$ realizes to a nonzero size, say s. In this case, $\widetilde{\sigma}$ would visit, u_{s}, the s-child of u and follows $T_{u_{\mathrm{s}}}$ from then on, but our algorithm visits, $l(v)_{\mathrm{s}}$, the s-child of the last node of that segment, and follows $T_{l(v)_{\mathrm{s}}}$. The expected profit loss in each such event can be bounded by $|\mathbb{P}(T_{u_{\mathrm{s}}}) - \mathbb{P}(T_{l(v)_{\mathrm{s}}})| \le \epsilon^5 \mathsf{OPT}$. Suppose $\widetilde{\sigma}$ pays such a profit loss, and switches to visit $l(v)_{\mathrm{s}}$. Hence, $\widetilde{\sigma}$ and our algorithm always stay at the same node. Note that the number of edges corresponding to nonzero size realizations is at most $\mathbb{C}/\epsilon^4$ in any root-to-leaf path. So $\widetilde{\sigma}$ pays at most $O(\epsilon^{-4})$ times in any realization. Therefore, the total profit loss is at most $O(\epsilon\mathsf{OPT})$.

Finally, we can show the probability that the auxiliary knapsack overflows is at most ϵ^8. Hence, the expected profit we gain is at least $(1-\epsilon^8)(1-O(\epsilon))\mathsf{OPT}$. □

5.2 Enumeration

We need to modify the dynamic program to incorporate the constraint that at most one item from each $\mathbb{B}_i$ can be packed. First, we assume that m is a polynomial of n. We will later show that we can also solve the problem even if m is exponentially large.

Now, we use $\mathrm{DP}(i, \mathsf{C}) = 1$ to denote the fact that we can reach configuration C using items from $\{\mathbb{B}_1, \ldots, \mathbb{B}_i\}$, such that in each realization path, at most one item from each $\mathbb{B}_i$ can appear at most once. Suppose we want to compute the value of $\mathrm{DP}(i, \mathsf{C})$ for some i and configuration C. Suppose we have computed the values of $\mathrm{DP}(i-1, \mathsf{C}')$ for all $\mathsf{C}' \leq \mathsf{C}$. We can decide to place items from $\mathbb{B}_i$ in a few blocks in the decision tree. The only constraint here is that we can place at most one item from $\mathbb{B}_i$ in each realization path. So the number of ways to do so is bounded by $m^{f(\epsilon)}$ for $f(\epsilon) = 2^{(1/\epsilon^5)^{O(1/\epsilon^{14})}}$, which is a polynomial of the input size. So the overall running time of the dynamic program is still a polynomial. The rest of the analysis is the same as before and we do not repeat it here. In summary, we have the following theorem, from which Theorem 1.6 follows as a direct corollary.

THEOREM 5.1. *For any $\epsilon > 0$, there is a polynomial time algorithm that finds a $(1+\epsilon)$-approximate adaptive policy for* GenSK *when the capacity is relaxed to $1+\epsilon$.*

We can also handle SK-CC even when the support of the size distribution of an item is of exponential size (represented implicitly). Combining Theorem 1.6 and the algorithm developed in [8] we can give a $(2+\epsilon)$-approximation for SK-Can (Theorem 1.7). This improves the factor 8 approximation algorithm developed in [23]. We note that the algorithm in [8] does not work for correlated sizes and profits. So whether there is $(2+\epsilon)$-approximation for SK-CC is still open. However, with mild assumptions on the size distributions, we can achieve an approximation factor of 2 for SK-CC. The details can be found in the full version.

6. BAYESIAN ONLINE SELECTION

In this section, we consider the *Bayesian online selection problem* subject to a knapsack constraint (denoted as BOSP-KC). Our problem falls into the framework formulated in [29]. In BOSP-KC, we are given a set of items $\mathbb{B} = \{b_1, b_2, \ldots, b_n\}$ and a knapsack capacity $\mathbb{C}$. Each item b_i has a random size X_i and a random profit P_i. X_i and P_i can be correlated but different items are independent of each other. The (discrete) joint distribution π of X_i and P_i is the input to the problem, in the form of $\pi_i(x,p) = \Pr(X_i = x, P_i = p)$. Let $\mathcal{D}_i$ be the support of π_i, i.e., $\mathcal{D}_i = \{(x,p) \mid \pi_i(x,p) \neq 0\}$. An adaptive policy σ can choose an item $b_i \in \mathbb{B}$ each time, view the size realization of X_i, and then make a irrevocable decision whether to pack b_i or not. If σ decides to pack b_i into the knapsack (given the remaining capacity is sufficient), the profit P_i is collected. Otherwise, no profit is collected, the remaining capacity does not change and we can not recall b_i later. If the items arrive in a predetermined order, we call the problem the *fixed order* BOSP-KC problem.

Our problem is also closely connected to the *knapsack secretary problem* [3] in the following sense. In the knapsack secretary problem, the size and the profit of each item are unknown in advance and the items arrive in a random order. When an item arrives, its size and profit become known to the decision maker and an irrevocable decision has to be made. Therefore, if we intentionally forget about the stochastic information and process the items in a random order, BOSP-KC becomes exactly the knapsack secretary problem, for which a constant factor competitive algorithm is known [3] [3].

In the remainder of this section, we focus on proving Theorem 1.8. In fact, we can reduce the problem to GenSK introduced in Section 5. Suppose we decide to consider b_i at a particular stage, and decide to accept b_i if and only if the realization $(X_i, P_i) \in D$ for some $D \subseteq \mathcal{D}_i$. We call D the *acceptance set*. Then, b_i is equivalent to an item with the size X_i^D and profit P_i^D, jointly distributed as follows:

$$(X_i^D, P_i^D) = \begin{cases} (X_i, P_i), & (X_i, P_i) \in D; \\ (0,0), & \text{Otherwise.} \end{cases}$$

However, the GenSK instance created can be exponential in size since each subset $D \subseteq \mathcal{D}_i$ corresponds to a distinct item in $\mathbb{B}_i$. We may reduce the size of $\mathbb{B}_i$ by exploiting the simple observation that in any optimal policy, if $(x,p) \in D$, any (x',p') with $x' \leq x$ and $p' \geq p$ (a realization with a smaller size and a larger profit) must be in D. But the resulting size is still exponential. However, this observation, combined with a discretization trick, can reduce the number of acceptance sets to a polynomial, while losing a profit of at most $O(\epsilon\mathsf{OPT})$.

Fixed order BOSP-KC: For the fixed order model, the above reduction also works. The only change is that in the GenSK instance we reduce to, we are required to examine the item classes $\mathbb{B}_i$ in a fixed order. Lemma 4.2 and Lemma 4.3 also hold in this case. We can also modify the dynamic program in Section 5.2 to find a block-adaptive policy subject to the order constraint. In fact, we can also get a constant competitive algorithm when compared with the offline optimum, using simple LP techniques [4]. The algorithm can provide the information of OPT, up to a constant factor, that is needed in the discretization.

7. CONCLUDING REMARKS

We develop the Poisson approximation technique and successfully apply it in finding approximate solutions for a variety of stochastic combinatorial optimization problems. These problems range from fixed set optimization problems to adaptive online optimization problems, from problems with a single probabilistic objective function, to problems with many probabilistic constraints. Our technique is conceptually simple, easy to apply, and has led to simplifications, generalizations and/or improvements of several previous results.

In the realm of approximating the distributions of the sums of n random variables, the Poisson approximation theorem and its relatives [6] work most effectively in the regime where, roughly speaking, the sum of the expected values of those random variables stays constant as n increases. This is quite different from the Gaussian approximations (e.g., CLT, Chernoff bounds, Berry-Esseen type inequalities) which typically require the sum of the expected values increases as n.

[3] We note that most work on secretary problems and prophet inequalities measures the performance of the algorithm by comparing the solution found by the online algorithm against the offline optimum.

[4] We can call such result a *knapsack prophet inequality*.

In fact, the theory of Poisson approximation is an important area in probability theory [6], but it has not been explored and utilized in algorithmic applications as extensively. We believe our technique, and more generally the theory of Poisson approximation, can find wider applications in stochastic combinatorial optimization and other algorithmic domains.

Acknowledgements

We would like to thank Yinyu Ye and Uri Zwick for stimulating discussions. This work was supported in part by the National Natural Science Foundation of China Grant 61202009, 61033001, 61061130540, 61073174, the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301.

8. REFERENCES

[1] S. Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. In *FOCS*, 2011.

[2] A. Armon and U. Zwick. Multicriteria global minimum cuts. *Algorithmica*, 46(1):15–26, 2006.

[3] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. A knapsack secretary problem with applications. *APPROX*, 2007.

[4] M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA*, 2007.

[5] N. Bansal, A. Gupta, J. Li, J. Mestre, V. Nagarajan, and A. Rudra. When lp is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

[6] A.D. Barbour, Lars Holst, and Svante Janson. *Poisson Approximation*.

[7] J.F. Bard and J.E. Bennett. Arc reduction and path preference in stochastic acyclic networks. *Management Science*, 37(2):198–215, 1991.

[8] A. Bhalgat. A (2+ ϵ)-approximation algorithm for the stochastic knapsack problem. 2012. manuscript.

[9] A. Bhalgat, A. Goel, and S. Khanna. Improved approximation results for stochastic knapsack problems. In *SODA*, 2011.

[10] P.M. Camerini, G. Galbiati, and F. Maffioli. Random pseudo-polynomial algorithms for exact matroid problems. *Journal of Algorithms*, 13(2):258–273, 1992.

[11] S. Chakraborty and O. Lachish. Improved competitive ratio for the matroid secretary problem. In *SODA*, 2012.

[12] S. Chawla, J.D. Hartline, D.L. Malec, and B. Sivan. Multi-parameter mechanism design and sequential posted pricing. In *STOC*, 2010.

[13] C. Chekuri and S. Khanna. On multi-dimensional packing problems. In *SODA*, 1999.

[14] C. Chekuri and S. Khanna. A PTAS for the multiple knapsack problem. In *SODA*, 2000.

[15] C. Chekuri, J. Vondrák, and R. Zenklusen. Multi-budgeted matchings and matroid intersection via dependent rounding. In *SODA*, 2011.

[16] C. Daskalakis and C. Papadimitriou. Computing equilibria in anonymous games. In *FOCS*, 2007.

[17] B.C. Dean, M.X. Goemans, and J. Vondrák. Adaptivity and approximation for stochastic packing problems. In *SODA*, pages 395–404, 2005.

[18] B.C. Dean, M.X. Goemans, and J. Vondrak. Approximating the Stochastic Knapsack Problem: The Benefit of Adaptivity. *Mathematics of Operations Research*, 33(4), 2008.

[19] E.B.Dynkin. The optimum choice of the instant for stopping a markov process. *Sov. Math. Dokl*, 4, 1963.

[20] S. Geetha and KPK Nair. On stochastic spanning tree problem. *Networks*, 23(8):675–679, 1993.

[21] A. Goel and P. Indyk. Stochastic load balancing and related problems. In *FOCS*, page 579, 1999.

[22] V. Goyal and R. Ravi. Chance constrained knapsack problem with random item sizes. *ORL*, 2009.

[23] A. Gupta, R. Krishnaswamy, M. Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *FOCS*, 2011.

[24] M.T. Hajiaghayi, R. Kleinberg, and T. Sandholm. Automated online mechanism design and prophet inequalities. In *AAAI*, volume 22, page 58, 2007.

[25] S. Im and Y. Wang. Secretary problems: Laminar matroid and interval scheduling. In *SODA*, 2011.

[26] H. Ishii, S. Shiode, and T. Nishida Yoshikazu. Stochastic spanning tree problem. *Discrete Applied Mathematics*, 3(4):263–273, 1981.

[27] P. Jaillet, J.A. Soto, and R. Zenklusen. Advances on matroid secretary problems: Free order model and laminar case. *arXiv preprint arXiv:1207.1333*, 2012.

[28] J. Kleinberg, Y. Rabani, and É. Tardos. Allocating bandwidth for bursty connections. In *STOC*, 1997.

[29] R. Kleinberg and S.M. Weinberg. Matroid prophet inequalities. In *STOC*, 2012.

[30] U. Krengel and L. Sucheston. Semiamarts and finite values. *Bull. Amer. Math. Soc*, 83:745–747, 1977.

[31] L. Le Cam. An approximation theorem for the poisson binomial distribution. *Pacific Journal of Mathematics*, 10(4):1181–1197, 1960.

[32] J. Li and A. Deshpande. Maximizing expected utility for stochastic combinatorial optimization problems. In *FOCS*, 2011.

[33] I. Murthy and S. Sarkar. Exact algorithms for the stochastic shortest path problem with a decreasing deadline utility function. *European Journal of Operational Research*, 103(1):209–229, 1997.

[34] I. Murthy and S. Sarkar. Stochastic shortest path problems with piecewise-linear concave utility functions. *Management Science*, 44(11):125–136, 1998.

[35] E. Nikolova. Approximation Algorithms for Reliable Stochastic Combinatorial Optimization. *APPROX*, 2010.

[36] E. Nikolova, M. Brand, and D.R. Karger. Optimal route planning under uncertainty. In *ICAPS*, 2006.

[37] E. Nikolova, J. Kelner, M. Brand, and M. Mitzenmacher. Stochastic shortest paths via quasi-convex maximization. In *ESA*, 2006.

[38] C.H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *FOCS*, 2000.

[39] R. Ravi and MX Goemans. The Constrained Minimum Spanning Tree Problem. In *SWAT*, 1996.