# Approximation Algorithms for Stochastic $k$-TSP*

**Alina Ene[1], Viswanath Nagarajan[2], and Rishi Saket[3]**

1 **Computer Science Department, Boston University.** aene@bu.edu
2 **Industrial and Operations Engineering Department, University of Michigan.** viswa@umich.edu
3 **IBM Research India.** rissaket@in.ibm.com

## Abstract

This paper studies the stochastic variant of the classical $k$-TSP problem where rewards at the vertices are independent random variables which are instantiated upon the tour's visit. The objective is to minimize the expected length of a tour that collects reward at least $k$. The solution is a policy describing the tour which may (*adaptive*) or may not (*non-adaptive*) depend on the observed rewards. Our work presents an adaptive $O(\log k)$-approximation algorithm for Stochastic $k$-TSP, along with a non-adaptive $O(\log^2 k)$-approximation algorithm which also upper bounds the *adaptivity gap* by $O(\log^2 k)$. We also show that the adaptivity gap of Stochastic $k$-TSP is at least $e$, even in the special case of stochastic knapsack cover.

**1998 ACM Subject Classification** F.2.2 Nonnumerical Algorithms and Problems

**Keywords and phrases** Stochastic TSP, algorithms, approximation, adaptivity gap

**Digital Object Identifier** 10.4230/LIPIcs.FSTTCS.2017.27

## 1 Introduction

In this paper, we consider the following stochastic variant of the classical $k$-TSP problem. The input consists of a metric $(V, d)$ with depot $r \in V$. Each vertex $v \in V$ has an independent stochastic reward $R_v \in \mathbb{Z}_+$. All reward distributions are given as input and thus they are known upfront, but the actual reward instantiation $R_v$ is only known when vertex $v$ is visited. Given a target value $k$, the goal is to find an adaptive tour originating from $r$ that collects a total reward at least $k$ with the minimum expected length.[1] We assume that all rewards are supported on $\{0, 1, \ldots, k\}$.

Any feasible solution to this problem can be described by a decision tree where nodes correspond to vertices that are visited and branches correspond to observed reward instantiations. The size of such decision trees can be exponentially large. So we focus on obtaining solutions (aka policies) that implicitly specify decision trees. These solutions are called *adaptive* because the choice of the next vertex to visit depends on past random instantiations.

We will also consider the special class of *non-adaptive* solutions. Such a solution is described simply by an ordered list of vertices: the policy involves visiting vertices in the given order until the target of $k$ is met (at which point the tour returns to $r$). These solutions are often preferred over adaptive solutions as they are easier to implement in practice. However, the performance of non-adaptive solutions may be much worse, and so it is important to bound the *adaptivity gap* [15] which is the worst-case ratio between optimal

* This work was partly done when the authors were at the IBM T. J. Watson Research Center.
[1] If the total instantiated reward $\sum_{v \in V} R_v$ happens to be less than $k$, the tour ends after visiting all vertices.


adaptive and non-adaptive policies. This approach via non-adaptive policies has been very useful for a number of stochastic optimization problems, eg. [15, 19, 22, 5, 23, 24, 6, 25].

The Stochastic $k$-TSP problem captures several well-studied problems, including the classical $k$-TSP problem where the rewards are deterministic, and the *stochastic knapsack cover* problem where the metric $(V, d)$ is a weighted star rooted at $r$: given $n$ items with each item $i \in [n]$ having deterministic cost $d_i$ and independent random reward $R_i \in \mathbb{Z}_+$, and a target $k$, find an adaptive policy that obtains total reward $k$ at the minimum expected cost. The $k$-TSP problem (and the related spanning tree variant $k$-MST) have received considerable attention, leading to the development of a 2-approximation algorithm for the problem [18]. The stochastic knapsack cover problem was considered recently in [16], where an adaptive 2-approximation algorithm was obtained; however, no bounds on the adaptivity gap were known previously even for this special case.

**Results and Techniques.** In this paper, we give the first approximation algorithm for the general stochastic $k$-TSP, with an approximation ratio of $O(\log k)$. We also show that the adaptivity gap is $O(\log^2 k)$. The adaptivity gap is constructive: we give a non-adaptive algorithm that is an $O(\log^2 k)$-approximation to the optimal adaptive policy.

▶ **Theorem 1.** *There is an adaptive algorithm for stochastic $k$-TSP with approximation ratio $O(\log k)$. Moreover, the adaptivity gap is upper bounded by $O(\log^2 k)$.*

To motivate our approach, consider an algorithm for the classical $k$-TSP problem which has access to a hypothetical[2] subroutine exactly solving the deterministic *orienteering* problem: given a metric $(V, d)$ on vertices with non-negative reward, a root vertex $r$, and a bound $t$ compute a tour starting at $r$ of length at most $t$ which collects the maximum reward. The algorithm iterates over tour lengths of increasing powers of 2, solving for each length the deterministic orienteering problem, until the union of the computed tours yields reward $k$. It can be seen that this is an $O(1)$-approximation to $k$-TSP. Our algorithm adapts this iterative method to Stochastic $k$-TSP using the expected *truncated* rewards at each vertex. At the beginning of each iteration, the rewards distribution at each vertex is appropriately truncated w.r.t. the residual target (and set to zero if the vertex has been previously visited). However, in the stochastic setting, even if we used an exact orienteering algorithm, we need to construct roughly $(\log k)$ tours for each length (see Example 2 in Section 2.1). We show that using $O(\log k)$ many tours of each length suffices to obtain an $O(\log k)$-approximation algorithm for stochastic $k$-TSP. Moreover, this multiple-tour approach also allows for the use of an $O(1)$-approximation for orienteering, which is needed for a poly-time algorithm.

The high-level idea in the analysis is to show that the non-completion probability in the algorithm drops (roughly) by a constant factor in each iteration. This approach is similar to that for the classic min-latency TSP [10, 17]. Such an approach has also been used for stochastic optimization problems in [27, 28], but those results do not apply directly to metric-based costs that we need to handle here. Our main technical contribution is in proving the bound on non-completion probabilities in successive iterations (Lemma 3). This involves upper and lower bounding the expected "relative gain" which is the expected (truncated) increase in reward divided by the residual target. See Section 2 for details.

The non-adaptive algorithm is based on "simulating" the above adaptive algorithm. Corresponding to each orienteering instance solved in the adaptive algorithm we now solve $\log_2 k$ different instances based on different power-of-two values for the residual target. This

[2] We do not expect a poly-time exact algorithm for orienteering as the problem is APX-hard [9].

results in a worse $O(\log^2 k)$ approximation ratio; this is relative to the optimal adaptive value and hence it bounds the adaptivity gap. We are not aware of a more direct approach for the non-adaptive problem, even if we compare to the optimal non-adaptive value.

We also show that the adaptivity gap is at least $e \approx 2.718$, which holds even in the special case of stochastic knapsack cover with a single random item.

▶ **Theorem 2.** *The adaptivity gap of* Stochastic $k$-TSP *(knapsack cover) is at least* $e$.

This result is obtained in an indirect manner. Using LP-duality we show that randomized online lower bounds for the *online bidding problem* [13] can be reduced to adaptivity gap lower bounds for stochastic knapsack-cover. These lower bound instances contain a *single* stochastic item. We can also show using a connection to the *incremental k-MST* problem [29], that the adaptivity gap for such instances is *exactly* $e$.

**Other related work.** Stochastic packing and covering problems have received considerable attention, leading to approximation guarantees and adaptivity gaps for several important problems, including knapsack [15, 8, 30, 16], packing and covering integer programs [14, 19], matching [12, 5, 1, 7, 2], matroid intersection [25, 3], submodular cover [20, 27, 21, 28], and orienteering [22, 24, 6]. Recently, [26] proved a constant adaptivity gap for submodular *maximization* problems under a very general class of constraints (including orienteering). This is however not directly applicable to stochastic $k$-TSP because we have (i) a minimization objective with strict covering requirement and (ii) general random variables as opposed to just binary in [26].

Sections 2 and 3 describe and analyze the adaptive and non-adaptive algorithms respectively for Stochastic $k$-TSP. Together they prove Theorem 1. The lower bound on the adaptivity gap (Theorem 2) is proved in Section 4.

## 2 Stochastic $k$-TSP Algorithm

Let us begin with an informal description of the adaptive algorithm. The algorithm relies on iteratively solving instances of the (deterministic) orienteering problem. In the orienteering problem, we are given a metric $(V, d)$, depot $r$, profits at vertices and a length bound $B$; the goal is to find a tour originating from $r$ of length at most $B$ that maximizes the total profit. There is a $(2+\epsilon)$-approximation algorithm for orienteering [11] which we can use directly. The vertex-profits in the orienteering instance are chosen to be truncated expectations of the random rewards $R_v$ where the truncation threshold is the current residual target. The length bounds are geometrically increasing with the phases of the algorithm. The algorithm terminates when the residual target reaches zero or all the vertices have been visited. However, in each phase the algorithm solves $\alpha \approx (\log k)$ many iterations each approximately solving the residual orienteering instance with the same length bound of that phase. This (roughly speaking) results in the $O(\log k)$ approximation ratio. The requirement of many iterations per phase turns out to be a somewhat subtle issue: we show in Section 2.1 that significantly reducing the number of repetitions results in a much worse approximation ratio.

The formal algorithm is given below. We assume (by scaling) that the minimum positive distance in the metric $(V, d)$ is one. At any point in the algorithm, $S$ denotes the set of currently visited vertices, $\sigma$ denotes the reward instantiations of $S$, and $k(\sigma)$ is the total observed reward. The number of iterations of the inner for-loop is $\alpha := c \cdot H_k$, where $c \geq 1$ is a constant to be fixed later and $H_k \approx \ln k$ is the $k$th harmonic number. We will show that Ad-kTSP is an $8\alpha$-approximation algorithm for Stochastic $k$-TSP.

**Algorithm 1** Algorithm Ad-kTSP

initialize $\sigma \leftarrow \emptyset$, $S \leftarrow \emptyset$ and $k(\sigma) = 0$.
**for** phase $i = 0, 1, \ldots$ **do**
  **for** $t = 1, \ldots, \alpha$ **do**
    define profits at vertices as follows

$$w_v := \begin{cases} \mathbb{E}[\min\{R_v, k - k(\sigma)\}] & \forall\, v \in V \setminus S \\ 0 & \forall\, v \in S \end{cases}$$

    using a $\rho$-approximation algorithm for the *orienteering* problem, compute a tour $\pi$ originating from $r$ of length at most $2^i$ with maximum total profit.
    traverse tour $\pi$ and observe the actual rewards; augment $S$ and $\sigma$ accordingly.
    if $k(\sigma) \geq k$ or all vertices have been visited, the solution ends.
  **end for**
**end for**

We view each iteration of the outer for loop as a *phase* and use $i$ to index the phases. For any phase $i \geq 0$, we define the following quantities:

$$u_i := \Pr[\text{Ad-kTSP continues beyond phase } i]$$
$$u_i^* := \Pr[\text{optimal policy continues beyond distance } 2^i]$$

Since the minimum distance in the metric is one we have $u_0^* = 1$.

### Overview of analysis

For analyzing the algorithm, observe that the expected cost of the obtained solution is roughly bounded by $\alpha \sum_{i \geq 0} u_i 2^{i+1}$ as the distance in phase $i$ is roughly $\alpha \cdot 2^i$. To bound this we show that (*) $u_i - u_{i^*} \leq u_{i-1}/4$ which allows us to relate the expected cost of the algorithm with the expected optimal cost yielding an $O(\alpha)$ approximation. To show (*), the first step (Lemma 4) is proving the existence, in the $i$th phase with "state" $\sigma$, of an orienteering solution of length at most $2^i$ with profit at least a fraction $p_{i^*}(\sigma) := (1 - u_{i^*}|_\sigma)$ of the current residual target. Here $u_{i^*}|_\sigma$ corresponds to the probability $u_{i^*}$ *conditioned* on the current state $\sigma$. Combined with a known inequality (Theorem 6) on the truncated sum of independent random variables, this is used to lower bound the expected *gain* (fraction of the residual target covered) of the adaptive algorithm in each phase by $\Omega(\alpha) \cdot (u_i - u_{i^*})$; see Claim 8. The expected gain on the other hand can easily be upper bounded by $H_k \cdot u_{i-1}$ where $H_k \approx \ln k$ is the $k$th harmonic number (Claim 7). Finally, setting $\alpha = \Theta(\log k)$ with a suitable constant finishes the proof.

Now to the formal details. The following lemma is the main component of the analysis.

▶ **Lemma 3.** *For any phase $i \geq 1$, we have $u_i \leq \frac{u_{i-1}}{4} + u_i^*$.*

Using Lemma 3, we can finish the analysis as follows. Let ALG denote the expected length of the tour constructed by the Ad-kTSP algorithm. Let OPT denote the expected length of the optimal adaptive tour. The total distance traveled by the Ad-kTSP algorithm in the first $i$ phases is at most $\alpha \sum_{j=0}^{i} 2^j \leq 2^{i+1}\alpha$. Using this we obtain $\text{ALG} \leq 2\alpha \sum_{i \geq 1} 2^i u_i + 4\alpha$. Also, $\text{OPT} \geq \frac{1}{2} \sum_{i \geq 1} 2^i u_i^* + u_0^* = \frac{1}{2} \sum_{i \geq 1} 2^i u_i^* + 1$. Letting

$T := \sum_{i\geq 1} 2^i u_i$ and using Lemma 3, we obtain

$$T \;\leq\; \frac{1}{4}\sum_{i\geq 1} 2^i \cdot u_{i-1} + \sum_{i\geq 1} 2^i \cdot u_i^* \;\leq\; \frac{1}{2}\sum_{i\geq 1} 2^i \cdot u_i + \sum_{i\geq 1} 2^i \cdot u_i^* + \frac{1}{2} \;\leq\; \frac{1}{2}T + 2\cdot \mathrm{OPT} - \frac{3}{2}.$$

It follows that $T \leq 4\cdot\mathrm{OPT}-3$ and $\mathrm{ALG} \leq 8\alpha\cdot\mathrm{OPT}$. Thus we obtain an $8\alpha$-approximation algorithm, which proves the first part of Theorem 1.

Now we turn to the proof of Lemma 3. We start by introducing some notation. Consider an arbitrary point in the execution of algorithm Ad-kTSP, and let $\langle S, \sigma, k(\sigma)\rangle$ be a triple in which $S$ is the set of vertices visited so far, $\sigma$ is the observed instantiation of $S$, and $k(\sigma)$ is the total reward observed in $S$. We refer to such a triple $\langle S, \sigma, k(\sigma)\rangle$ as the *state* of the algorithm.

▶ **Lemma 4.** *Consider a state $\langle S, \sigma, k(\sigma)\rangle$ of the* Ad-kTSP *algorithm. Consider the* Orienteering *instance in which each vertex in $S$ is assigned a profit of zero and each vertex $v$ not in $S$ is assigned a profit of $\mathbb{E}[\min\{R_v, k-k(\sigma)\}]$. There is an orienteering tour from $r$ of length at most $2^i$ with profit at least $(k-k(\sigma))\cdot p_i^*(\sigma)$, where*

$$p_i^*(\sigma) = \Pr\left[\text{ optimal policy completes before distance } 2^i \mid \sigma\right].$$

**Proof.** Consider the tree $\mathcal{T}^*$ representing the optimal policy. We *condition* on the instantiations $\sigma$ on vertices $S$ to obtain tree $\mathcal{T}^*(S,\sigma)$. Formally, we start with $\mathcal{T}^*(S,\sigma) = \mathcal{T}^*$ and apply the following transformation for each vertex $v \in S$ with instantiation $\sigma_v$: at each node $\nu$ in $\mathcal{T}^*(S,\sigma)$ corresponding to $v$, we remove all subtrees of $\nu$ except the subtree corresponding to instantiation $\sigma_v$; additionally, the probability of this edge (labeled $\sigma_v$) is set to one. Note that the probabilities at nodes corresponding to vertices $V \setminus S$ are unchanged, since rewards at different vertices are independent.

Now, *mark* those nodes in $\mathcal{T}^*(S,\sigma)$ that correspond to completion (i.e. reward at least $k$ is collected) within distance $2^i$. Note that the probability of reaching a marked node is exactly $p_i^*(\sigma)$. Finally, let $\mathcal{T}$ denote the subtree of $\mathcal{T}^*(S,\sigma)$ containing only those nodes that have a marked descendant. When tree $\mathcal{T}$ is traversed, the probability of reaching a leaf-node is $p_i^*(\sigma)$; with the remaining probability the traversal ends at some internal node. Clearly, every leaf-node in $\mathcal{T}$ is marked and hence corresponds to an $r$-tour of length at most $2^i$ along with instantiated rewards that sum to at least $k$. Define modified rewards at nodes of $\mathcal{T}$ as follows:

$$\hat{R}_v = \begin{cases} \min\{R_v, k-k(\sigma)\} & \text{if } v \notin S \\ 0 & \text{if } v \in S \end{cases}$$

Notice that the profits in the deterministic Orienteering instance are precisely $w_v = \mathbb{E}[\hat{R}_v]$. Observe that the sum of modified rewards at each leaf of $\mathcal{T}$ is at least $k-k(\sigma)$. Thus the expected $\hat{R}$-reward obtained in $\mathcal{T}$ is $\hat{E} \geq (k-k(\sigma))\cdot p_i^*(\sigma)$. Also,

$$\hat{E} = \sum_{\nu\in\mathcal{T}} \Pr[\text{reach } \nu]\cdot\mathbb{E}\left[\hat{R}_\nu \mid \text{reach } \nu\right] = \sum_{\nu\in\mathcal{T}} \Pr[\text{reach } \nu]\cdot\mathbb{E}\left[\hat{R}_\nu\right] = \sum_{\nu\in\mathcal{T}} \Pr[\text{end at } \nu]\cdot\left(\sum_{\mu\preceq\nu} w_\mu\right).$$

Above, for a node $\nu \in \mathcal{T}$, we use $\hat{R}_\nu$ and $w_\nu$ to denote the respective variables for the vertex (in $V$) corresponding to $\nu$. Also, the notation $\mu \preceq \nu$ refers to node $\mu$ being an ancestor of node $\nu$ in $\mathcal{T}$. The first equality is by definition of $\hat{E}$, the second uses the fact that $\{\hat{R}_v : v \in V\}$ are independent, and the last is an interchange of summation. By averaging, there is some node $\nu' \in \mathcal{T}$ with $\sum_{\mu\prec\nu'} w_\mu \geq \hat{E} \geq (k-k(\sigma))\cdot p_i^*(\sigma)$. We can assume that $\nu'$

is a leaf node (otherwise we can reset $\nu'$ to be any leaf node of $\mathcal{T}$ below $\nu'$). So the $r$-tour corresponding to this node $\nu'$ has length at most $2^i$ and is feasible for the deterministic ORIENTEERING instance. Moreover, it has profit at least $(k - k(\sigma)) \cdot p_i^*(\sigma)$ as claimed. ◀

▶ **Lemma 5.** *Consider a state $\langle S, \sigma, k(\sigma)\rangle$ of the* AD-KTSP *algorithm with $k(\sigma) < k$. Consider the* ORIENTEERING *instance with profits $\{w_v : v \in V\}$ where $w_v = 0$ for $v \in S$ and $w_v = \mathbb{E}[\min\{R_v, k - k(\sigma)\}]$ for $v \notin S$. Let $\pi$ be any $\rho$-approximate orienteering tour consisting of vertices $V(\pi)$. Then,*

$$\mathbb{E}\left[\min\left\{\sum_{v\in V(\pi)\setminus S} R_v,\, k - k(\sigma)\right\}\right] \quad \geq \quad \frac{1}{\rho}\cdot\left(1-\frac{1}{e}\right)\cdot(k-k(\sigma))\cdot p_i^*(\sigma).$$

**Proof.** For each $v \in V(\pi)\setminus S$ define $X_v := \min\left\{\frac{R_v}{k-k(\sigma)}, 1\right\}$. Note that $X_v$s are independent $[0,1]$ random variables, and $\mathbb{E}[X_v] = \frac{w_v}{k-k(\sigma)}$. Let $X := \sum_{v\in V(\pi)\setminus S} X_v$ and $Y = \min(X, 1)$. So the profit obtained by solution $\pi$ to the deterministic orienteering instance is $(k - k(\sigma)) \cdot \mathbb{E}[X]$. Using Lemma 4 and the fact that $\pi$ is a $\rho$-approximate solution, we obtain $\mathbb{E}[X] \geq \frac{1}{\rho}\cdot p_i^*$. We can now complete the proof of the lemma by applying Theorem 6 below. ◀

▶ **Theorem 6** ([4])**.** *Given a set $\{X_v\}$ of independent $[0,1]$ random variables with $X = \sum X_v$ and $Y = \min(X, 1)$, we have $\mathbb{E}[Y] \geq (1 - 1/e) \cdot \min\{\mathbb{E}[X], 1\}$.*

Now we are ready to prove Lemma 3.

**Proof of Lemma 3.** Consider any phase $i \geq 1$ and let $\langle S, \sigma, k(\sigma)\rangle$ be the state of the AD-KTSP algorithm at the start of some iteration of the inner loop. Let $\pi$ be the orienteering tour that the algorithm visits next. Define

$$\text{gain}(\langle S, \sigma\rangle) := \frac{\mathbb{E}\left[\min\left\{\sum_{v\in V(\pi)\setminus S} R_v,\, k - k(\sigma)\right\}\right]}{k - k(\sigma)} \quad \text{if } k(\sigma) < k,$$

and $\text{gain}(\langle S, \sigma\rangle) := 0$ if $k(\sigma) \geq k$. The quantity $\text{gain}(\langle S, \sigma\rangle)$ measures the expected fraction of the residual target that we cover after visiting $\pi$.

Let $I(\sigma) = [k(\sigma) \geq k]$ be an indicator variable that is equal to one if $k(\sigma) \geq k$ and it is equal to zero otherwise. We have:

$$\text{gain}(\langle S, \sigma\rangle) \quad \geq \quad \frac{1}{\rho}\cdot\left(1-\frac{1}{e}\right)\cdot\left(p_i^*(\sigma) - I(\sigma)\right). \tag{1}$$

To see this, note that if $I(\sigma)$ is one, the gain is zero and the inequality above holds since $p_i^*(\sigma) \leq 1$; and if $I(\sigma)$ is zero, this inequality follows from Lemma 5.

Recall that there are $\alpha$ iterations of the inner loop in phase $i$, and we use $t \in [\alpha]$ to index the iterations. For each iteration $t$ (of phase $i$), let $S_t$ be the random variable denoting the vertices visited until iteration $t$, and let $\sigma_t$ denote the reward instantiations of $S_t$. Define:

$$G_t \quad := \quad \mathop{\mathbb{E}}_{\langle S_t, \sigma_t\rangle}[\text{gain}(\langle S_t, \sigma_t\rangle)]$$

Let $\Delta = \sum_{t=1}^{\alpha} G_t$ denote the total gain in phase $i$. The proof relies on upper and lower bounding $\Delta$, which is done in the next two claims.

▶ **Claim 7.** *We have $\Delta \leq H_k \cdot u_{i-1}$.*

**Proof.** Let $\Delta(q)$ denote the value of $\Delta$ conditioned on the instantiations $q$ of all random variables. If Ad-kTSP (conditioned on $q$) finishes before phase $i$ then gain in each iteration is zero and $\Delta(q) = 0$.

In the following, we assume that Ad-kTSP (conditioned on $q$) reaches phase $i$. Let $L \leq k$ denote the residual target at the start of phase $i$, and $J_1, \ldots, J_\alpha \in \mathbb{Z}_+$ the incremental rewards obtained in each of the $\alpha$ iteration of phase $i$; recall that all rewards are integral. Then,

$$\Delta(q) \;\leq\; \sum_{t=1}^{\alpha} \frac{J_t}{L - J_1 - \cdots - J_{t-1}} \;\leq\; \sum_{t=1}^{L} \frac{1}{t} \;=\; H_L \;\leq\; H_k$$

The claim now follows since the algorithm reaches phase $i$ only with probability $u_{i-1}$. ◀

▶ **Claim 8.** *We have* $\Delta \geq \frac{\alpha}{\rho} \cdot \left(1 - \frac{1}{e}\right) \cdot (u_i - u_i^*)$.

**Proof.** For any $t \in [\alpha]$, by Inequality (1), we have

$$G_t \;=\; \mathop{\mathbb{E}}_{\langle S_t, \sigma_t \rangle}[\text{gain}(\langle S_t, \sigma_t \rangle)] \;\geq\; \frac{1}{\rho} \cdot \left(1 - \frac{1}{e}\right) \cdot \mathop{\mathbb{E}}_{\sigma_t}[p_i^*(\sigma_t) - I(\sigma_t)]$$

Let $p(t)$ be the probability that Ad-kTSP finishes by iteration $t$ of phase $i$. Since the probabilities $p(t)$ are non-decreasing in $t$, we have $p(t) \leq p(\alpha) = \Pr[\text{Ad-kTSP finishes by phase } i] = 1 - u_i$. For a fixed iteration $t$, the possible outcomes of $\langle S_t, \sigma_t \rangle$ correspond to a partition of the overall sample space. So we have $\mathbb{E}_{\sigma_t}[I(\sigma_t)] = p(t) \leq 1 - u_i$ and $\mathbb{E}_{\sigma_t}[p_i^*(\sigma_t)] = \Pr[\text{optimal policy completes within distance } 2^i] = 1 - u_i^*$. This completes the proof since $\Delta = \sum_{t=1}^{\alpha} G_t$. ◀

It follows from the two claims that

$$\frac{\alpha}{\rho} \cdot (1 - 1/e) \cdot (u_i - u_i^*) \;\leq\; \Delta \;\leq\; H_k \cdot u_{i-1}.$$

Setting $\alpha = 4\rho \frac{e}{e-1} \cdot H_k$ implies the lemma. ◀

We conclude this section by showing that our analysis of Ad-kTSP is essentially tight, and that it is necessary for $\alpha$ to be $\tilde{\Omega}(\log k)$.

## 2.1 Tightness of analysis of Ad-kTSP

Example 1. The analysis of Ad-kTSP is tight up to constant factors, even in the deterministic setting. Consider an instance of deterministic knapsack cover with $k = 2^\ell$ (for large integer $\ell$) and $\ell(\ell+1)$ items as follows. For each $i \in \{0, 1, \ldots, \ell\}$ there is one item of cost $2^i$ and reward $2^i$, and $\ell - 1$ items of cost $2^i$ and reward 1. Clearly the optimal cost is $2^\ell$. However the algorithm will select in each iteration $i = 0, 1, \ldots, \ell - 3$, all $\ell$ items of cost $2^i$: note that the reward from these items is at most $\ell^2 + \sum_{i=0}^{\ell-3} 2^i \leq \ell^2 + 2^{\ell-2} < 2^\ell$. So the algorithm's cost is at least $\ell \cdot 2^{\ell-3}$.

Example 2. A natural variant of Ad-kTSP is to perform the inner iterations (for each phase $i = 0, 1, \ldots$) a constant number of times instead of $\Theta(\log k)$. Next, we show an example where such variants perform poorly even with access to a hypothetical subroutine solving the deterministic orienteering problem exactly. It is worth noting that in the deterministic case, such an approach (using an exact orienteering algorithm *once* for each phase $i$) suffices to obtain an $O(1)$-approximation for $k$-TSP. However, if we used an $O(1)$-approximation for orienteering, then such an approach performs poorly even for the deterministic $k$-TSP.

Consider the variant of Ad-kTSP that performs $1 \le h = o\left(\frac{\log k}{\log\log k}\right)$ iterations in each phase $i$. Choose $t \in \mathbb{Z}_+$ and set $\delta = 1/(ht)$ and $k = (ht)^{2ht} \in \mathbb{Z}_+$. Note that $ht = \Theta(\log k/\log\log k)$. Define a star metric centered at the depot $r$, with $ht+1$ leaves $\{u_{ij} : 0 \le i \le t-1, 0 \le j \le h-1\} \bigcup \{w\}$. The distances are $d(r,u_{ij}) = 2^i$ for all $i \in [t]$ and $j \in [h]$; and $d(r,w) = 1$. Each $u_{ij}$ contains three co-located items: one of deterministic reward $(1-\delta)\delta^{hi+j}\cdot k$, and two having reward $\delta^{hi+j}\cdot k$ with probability $\delta$ (and zero otherwise). Finally $w$ contains a deterministic item of reward $k$. By the choice of parameters, all rewards are integer valued. The optimal cost is 2, just visiting vertex $w$.

Now consider the execution of the modified Ad-kTSP algorithm. The probability that all the random items in the $u_{ij}$-vertices have zero reward is $(1-\delta)^{2ht} \ge \Omega(1)$. Conditioned on this event, it can be seen inductively that in the $j^{th}$ iteration of phase $i$ (for all $i$ and $j$),

- the total observed reward until this point is $k(1-\delta)\sum_{\ell=0}^{hi+j-1}\delta^\ell = k\left(1-\delta^{hi+j}\right)$.
- the algorithm's tour (and optimal solution to the orienteering instance) involves visiting just vertex $u_{ij}$ and choosing the three items in $u_{ij}$, for a total profit of $(1+\delta)\cdot\delta^{hi+j}\cdot k$.

Thus the expected cost of this algorithm is $\Omega(h\cdot 2^t)$, implying an approximation ratio $\exp\left(\frac{\log k}{h\cdot\log\log k}\right)$.

## 3 Non-Adaptive Algorithm

For our non-adaptive algorithm we show that the previous adaptive policy can be simulated in a non-adaptive manner, resulting in an $O(\log^2 k)$-approximate non-adaptive algorithm. This also upper bounds the adaptivity gap by the same quantity, proving the second part of Theorem 1. The difference from the adaptive setting is that the non-adaptive algorithm does not know the reward accrued so far and thus tries many possible residual targets for the orienteering problem. Specifically, at each of the $\alpha$ iterations in a phase, we append $\ell \approx \log_2 k$ different solutions to the orienteering problem defined with residual targets $k/2^j, j = 0, 1, \ldots, \lfloor \log_2 k \rfloor$. The analysis is almost identical to that of the adaptive algorithm. As before, we set $\alpha = O(H_k)$, yielding a $O(\ell\alpha) = O(\log^2 k)$ approximation.

The formal algorithm NonAd-kTSP is given below. The non-adaptive tour $\tau$ is constructed iteratively; $S$ denotes the set of vertices visited in the current tour.

The difference from Ad-kTSP is in the inner for loop (indexed by $j$); this handles the fact that (unlike Ad-kTSP) the non-adaptive algorithm does not know the reward accrued so far. We set $\alpha := c'\cdot H_k$, where $c'$ is a constant to be fixed later. Let $\ell := 1 + \lfloor \log_2 k \rfloor$ the number of iterations of the innermost loop.

The analysis for NonAd-kTSP is almost identical to Ad-kTSP. We will show that this is an $(8\alpha\ell)$-approximation algorithm for Stochastic $k$-TSP. As before, each iteration of the outer for loop is called a *phase*, which is indexed by $i$. For any phase $i \ge 0$, define

$$u_i := \Pr[\text{NonAd-kTSP continues beyond phase } i]$$
$$u_i^* := \Pr[\text{optimal adaptive policy continues beyond distance } 2^i]$$

Just as in Lemma 3 we will show:

$$u_i \le \frac{u_{i-1}}{4} + u_i^*, \qquad \forall i \ge 1. \tag{2}$$

Since the total distance traveled by NonAd-kTSP in the first $i$ phases is at most $\ell\alpha\sum_{h=0}^{i} 2^h \le \ell\alpha\, 2^{i+1}$, it follows (as for Ad-kTSP) that the expected length ALG of NonAd-kTSP is at most $8\ell\alpha\cdot\mathrm{OPT}$.

**Algorithm 2** Algorithm NonAd-kTSP

initialize $\tau, S \leftarrow \emptyset$.
**for** phase $i = 0, 1, \ldots$ **do**
  **for** $t = 1, \ldots, \alpha$ **do**
    **for** $j = 0, 1, \ldots, \lfloor \log_2 k \rfloor$ **do**
      define profits as follows

$$w_v := \begin{cases} \mathbb{E}[\min\{R_v, k/2^j\}] & \text{for } v \in V \setminus S \\ 0 & \text{for } v \in S \end{cases}$$

      using a $\rho$-approximation algorithm for the *orienteering* problem, compute a tour $\pi$ originating from $r$ of length at most $2^i$ with maximum total profit.
      append tour $\pi$ to $\tau$, i.e. $\tau \leftarrow \tau \circ \pi$.
    **end for**
  **end for**
**end for**

We now prove (2). Consider a fixed phase $i \geq 1$ and one of the $\alpha$ iterations of the second for-loop (indexed by $t$). Let $S$ denote the set of vertices visited before the start of this iteration $\langle i, t \rangle$. Let $\sigma$ denote the reward instantiations at $S$, and $k(\sigma)$ the total reward in $\sigma$. Note that $\sigma$ is only used in the analysis and not in the algorithm. Let $V(i,t) \subseteq V \setminus S$ be the new vertices visited in iteration $\langle i, t \rangle$; these come from $\ell$ different subtours corresponding to the inner for-loop. We will show (analogous to Lemma 5) that:

$$\mathbb{E}\left[\min\left\{\sum_{v \in V(i,t)} R_v,\, k - k(\sigma)\right\}\right] \quad \geq \quad \frac{1}{2\rho} \cdot \left(1 - \frac{1}{e}\right) \cdot (k - k(\sigma)) \cdot p_i^*(\sigma). \tag{3}$$

Above, $p_i^*(\sigma) = \Pr\left[\text{ optimal adaptive policy completes before distance } 2^i \mid \sigma\right]$. Exactly as in the proof of Lemma 3, this would imply (2) when we set $\alpha = 8\rho\frac{e}{e-1} \cdot H_k$.

It remains to prove (3). Let $g \in \{0, 1, \ldots, \ell\}$ denote the index such that $\frac{k}{2^g} \leq k - k(\sigma) < \frac{k}{2^{g-1}}$ and let $c := k/2^g$. An identical proof to Lemma 4 implies:

▶ **Lemma 9.** *Consider the* Orienteering *instance with profits* $s_v := \mathbb{E}[\min\{R_v, c\}]$ *for* $v \in V \setminus S$ *and* $s_v := 0$ *otherwise. There is an orienteering tour of length at most* $2^i$ *with profit at least* $c \cdot p_i^*(\sigma)$.

Let $T \subseteq V \setminus S$ denote the vertices added in iteration $\langle i, t \rangle$ corresponding to inner loop iterations $j \in \{0, 1, \ldots, g-1\}$. Note that in the inner loop iteration $j = g$, we have profits $w_v = \mathbb{E}[\min\{R_v, c\}]$ for $v \in V \setminus (S \cup T)$ and $w_v = 0$ otherwise. Lemma 9 now implies that the optimal profit of the orienteering instance in iteration $j = g$ is at least $c \cdot p_i^*(\sigma) - \sum_{u \in T} \mathbb{E}[\min\{R_u, c\}]$. Let $T'$ denote the vertices added in the inner-loop iteration $j = g$. Since we obtain a $\rho$-approximate solution, it follows that the additional profit obtained $\sum_{v \in T'} w_v \geq \frac{1}{\rho}\left(c \cdot p_i^*(\sigma) - \sum_{u \in T} \mathbb{E}[\min\{R_u, c\}]\right)$. So,

$$\sum_{v \in V(i,t)} \mathbb{E}[\min\{R_v, c\}] \quad \geq \quad \sum_{v \in T \cup T'} \mathbb{E}[\min\{R_v, c\}] \quad \geq \quad \frac{c}{\rho} \cdot p_i^*(\sigma).$$

Exactly as in Lemma 5 we now obtain:

$$\mathbb{E}\left[\min\left\{\sum_{v\in V(\pi)\setminus S} R_v,\, c\right\}\right] \quad \geq \quad \frac{1}{\rho}\cdot\left(1-\frac{1}{e}\right)\cdot c\cdot p_i^*(\sigma).$$

And using $c \leq k - k(\sigma) < 2c$, we obtain (3).

In the following few paragraphs we give example showing that our analysis of NonAd-kTSP is tight, and that it is necessary for $h$ to be $\tilde{\Omega}(\log k)$.

## 3.1 Tightness of analysis of NonAd-kTSP

Example 3. The analysis of NonAd-kTSP is tight up to constant factors, even in the deterministic setting. This example is similar to that for Ad-kTSP. Consider an instance of deterministic knapsack cover with $k = 2^\ell$ and $\ell^2(\ell+1)$ items as follows. For each $i \in \{0, 1, \ldots, \ell\}$ there is one item of cost $2^i$ and reward $2^i$, and $\ell^2 - 1$ items of cost $2^i$ and reward 1. Clearly the optimal cost is $2^\ell$. However algorithm NonAd-kTSP will select in each iteration $i = 0, 1, \ldots, \ell-3$, all $\ell^2$ items of cost $2^i$: note that the reward from these items is at most $\ell^3 + \sum_{i=0}^{\ell-3} 2^i \leq \ell^3 + 2^{\ell-2} < 2^\ell$. So the algorithm's cost is at least $\ell^2 \cdot 2^{\ell-3}$, implying an approximation ratio of $\Omega(\log^2 k)$.

Example 4. One might also consider the variant of NonAd-kTSP where the second for-loop (indexed by $t$) is performed $1 \leq h \ll \log k$ times instead of $\Theta(\log k)$. The following example shows that such variants have a large approximation ratio. Consider an instance of stochastic knapsack cover with $k = 2^\ell$. There are an infinite number of items, each of cost one and reward $k$ with probability $\frac{2}{3\ell}$ (the reward is otherwise zero). For each $i \in \{0, 1, \ldots, \ell/h\}$ and $j \in \{1, \ldots, \ell\}$ there are $h$ items of cost $2^i$ and reward $k/2^j$ with probability $\frac{2}{3\ell}$ (the reward is otherwise zero). The optimal (adaptive and non-adaptive) policy considers only the cost one items with $\{0, k\}$ reward, and has optimal cost $O(\ell)$.

Algorithm NonAd-kTSP chooses in each iteration $i \in \{0, 1, \ldots, \ell/h\}$ (of the first for-loop), iteration $t \in \{1, \cdots, h\}$ (of the second for-loop) and iteration $j \in \{0, 1, \ldots, \ell\}$ (of the third for-loop) one of the items of cost $2^i$ with $\{0, k/2^j\}$ reward. (This is an optimal choice for the orienteering instance as rewards are capped at $k/2^j$ and the length bound is $2^i$.) These items have total cost $\Theta(\ell \cdot 2^{\ell/h})$. For each $j \in \{0, 1, \ldots, \ell\}$ there are $\frac{\ell}{h} \cdot h = \ell$ items having reward $k/2^j$ with probability $\frac{2}{3\ell}$. It can be seen that the total reward from these items is less than $k$ with constant probability. So the expected cost of NonAd-kTSP is $\Omega(\ell \cdot 2^{\ell/h})$, implying an approximation ratio $\Omega(2^{\ell/h})$.

# 4 Lower Bound on Adaptivity Gap

We show that the adaptivity gap of Stochastic $k$-TSP is at least $e \approx 2.718$ (Theorem 2). This holds even in the special case of the *stochastic covering knapsack* problem with a single random item. For such instances, we can even prove that the adaptivity gap is exactly $e$.

Here is an overview of the proof. We start by embedding an online bidding problem into a stochastic knapsack-cover instance such that non-adaptive (adaptive) policies for the latter correspond to online (offline) bidding strategies for the former. A result of Chrobak et al. [13] shows that no online bidding strategy is better than $e$-competitive. To complete the analysis, LP duality is used to show that the worst possible adaptivity gap for the stochastic knapsack-cover instance equals the best possible competitiveness of any online bidding strategy.

The gap example is based on the following problem studied in [13].

▶ **Definition 10** (Online Bidding Problem)**.** Given input $n \in \mathbb{Z}_+$, an algorithm outputs a randomized sequence $b_1, b_2, \ldots, b_\ell$ of bids from the set $[n] := \{1, 2, \ldots, n\}$. The algorithm's cost under (an unknown) threshold $T \in [n]$ equals the (expected) sum of its bid values until it bids a value at least $T$. The algorithm is $\beta$-competitive if its cost under threshold $T$ is at most $\beta \cdot T$, for all $T \in [n]$.

▶ **Theorem 11** ([13])**.** *There is no randomized algorithm for online bidding that is less than $e$-competitive.*

Without loss of generality, any bid sequence must be increasing; let $\Gamma$ denote the set of increasing sequences on $[n]$. For any $I \in \Gamma$ and $T \in [n]$, let $C(I, T)$ denote the cost of sequence $I$ under threshold $T$. In terms of this notation, Theorem 11 is equivalent to:

$$\min_{\pi\,:\,\text{distribution}(\Gamma)} \quad \max_{T \in [n]} \quad \frac{\mathbb{E}_{I \leftarrow \pi}[C(I,T)]}{T} \quad \geq \quad e. \tag{4}$$

We now define the stochastic covering knapsack instance. The target $k := 2^{n+1}$. There is one random item $r$ of zero cost having reward $k - 2^i$ with probability $p_i$ (for all $i \in [n]$). There are $n$ deterministic items $\{u_i\}_{i=1}^n$ where $u_i$ has cost $i$ and reward $2^i$. We will show that there exist probabilities $p_i$s so that the adaptivity gap is $e$.

It is clear that an optimal policy (adaptive or non-adaptive) will first choose item $r$, since it has zero cost. Moreover, an optimal adaptive policy will next choose item $u_i$ exactly when $r$ is observed to have reward $k - 2^i$. Hence the optimal adaptive cost is $\sum_{i=1}^n i \cdot p_i$.

Any non-adaptive policy is given by a sequence $\tau$ of the deterministic items; recall that item $r$ is always chosen first. Moreover, due to the exponentially increasing rewards, we can assume that $\tau$ is an increasing sub-sequence of $\{u_i : 1 \leq i \leq n\}$. So there is a one-to-one correspondence between non-adaptive policies and $\Gamma$ (in the online bidding instance). Note that the cost of non-adaptive solution $I \in \Gamma$ is exactly $\sum_{T=1}^n p_T \cdot C(I, T)$; if the random item $r$ has size $k - 2^T$ then the policy will keep choosing items in $I$ until it reaches an index at least $T$. Thus, the maximum adaptivity gap achieved by such an instance is:

$$\max_{p\,:\,\text{distribution}([n])} \quad \min_{I \in \Gamma} \quad \frac{\mathbb{E}_{T \leftarrow p}[C(I,T)]}{\mathbb{E}_{T \leftarrow p}[T]}. \tag{5}$$

The next lemma relates the quantities in (4) and (5). Below, for any set $S$, $\mathcal{D}(S)$ denotes the collection of probability distributions on $S$.

▶ **Lemma 12.** *For any non-negative matrix $C(\Gamma, [n])$, we have:*

$$\max_{p \in \mathcal{D}([n])} \min_{I \in \Gamma} \frac{\mathbb{E}_{T \leftarrow p}[C(I,T)]}{\mathbb{E}_{T \leftarrow p}[T]} \quad = \quad \min_{\pi \in \mathcal{D}(\Gamma)} \max_{T \in [n]} \frac{\mathbb{E}_{I \leftarrow \pi}[C(I,T)]}{T}.$$

**Proof.** The second expression equals the LP:

$$\begin{array}{lll} \min & \beta & \\ \text{s.t.} & T \cdot \beta - \sum_{I \in \Gamma} C(I,T) \cdot \pi(I) \geq 0, & \forall T \in [n], \\ & \sum_{I \in \Gamma} \pi(I) \geq 1, & \\ & \beta, \pi \geq 0. & \end{array}$$

Taking the dual, we obtain:

$$\begin{array}{lll} \max & \alpha & \\ \text{s.t.} & \alpha - \sum_{T \in [n]} C(I,T) \cdot \sigma(T) \leq 0, & \forall I \in \Gamma, \\ & \sum_{T \in [n]} T \cdot \sigma(I) \leq 1, & \\ & \alpha, \sigma \geq 0. & \end{array}$$

Define functions $g, f : [n] \to \mathbb{R}_+$ where

$$f(p) := \min_{I \in \Gamma} \sum_{T=1}^{n} p_T \cdot C(I,T) \quad \text{and} \quad g(p) := \sum_{T=1}^{n} p_T \cdot T.$$

Note that when $p$ corresponds to a probability distribution, $f(p) = \min_{I \in \Gamma} \mathbb{E}_{T \leftarrow p}[C(I,T)]$ and $g(p) = \mathbb{E}_{T \leftarrow p}[T]$. So the first expression in the lemma is just $\max_{p \in \mathcal{D}([n])} f(p)/g(p)$. The key observation is that $f$ and $g$ are homogeneous, i.e. $f(a \cdot p) = a \cdot f(p)$ and $g(a \cdot p) = a \cdot g(p)$ for any scalar $a \in \mathbb{R}_+$ and vector $p \in \mathbb{R}^n_+$. This implies that the first expression equals:

$$\max_{p \in \mathcal{D}([n])} \frac{f(p)}{g(p)} \quad = \quad \max_{p \in \mathbb{R}^n_+} \frac{f(p)}{g(p)} \quad = \quad \max_{p \in \mathbb{R}^n_+ \,:\, g(p) \le 1} f(p).$$

It is easy to check that this equals the dual LP above, which proves the lemma. ◀

Thus the above instance of stochastic knapsack cover has adaptivity gap at least $e$, which proves Theorem 2. It can also be shown that every instance of stochastic knapsack cover with a single stochastic item has adaptivity gap at most $e$: this uses a relation to the incremental $k$-MST problem [29].

▶ **Proposition 13.** *The adaptivity gap of any instance of stochastic knapsack cover with a single random item is at most $e$.*

**Proof.** Consider any such instance with item $n$ having random reward $R \in \{0, 1, \cdots k\}$ and each item $i \in [n-1]$ having deterministic reward $r_i$. Let $c_i$ denote the cost of each $i \in [n]$. As there is only one random item, any adaptive policy is of the following form (i) select a subset $T_1 \subseteq [n-1]$ of deterministic items, (ii) select random item $n$, (iii) depending on the value of reward $R$, select subset $T_2(R) \subseteq [n-1]$ of deterministic items.

Note that if the reward in $T_1$ is itself at least $k$ then we do not even need to use the random item: so the optimal adaptive and non-adaptive policies are the same (both are $T_1$). On the other hand, if $r(T_1) < k$ then the random item will be selected with probability one: so we can assume that it is the first item to be selected, i.e. $T_1 = \emptyset$.

Now, consider the following *incremental $k$-MST* instance. There are $n$ vertices with distances induced by a star with root $n$ and leaves $[n-1]$ where the distance $d(n,i) = c_i$. There is zero reward at the root and reward $r_i$ at each $i \in [n-1]$. The goal is to find a random sequence $\sigma$ of edges so that the following ratio is minimized:

$$\max_{\ell \ge 0} \quad \frac{\mathbb{E}_\sigma[\text{cost of minimal prefix of } \sigma \text{ with reward at least } \ell]}{\text{optimal } \ell\text{-MST value}}.$$

As shown in [29], there is always a random sequence $\sigma$ with the above ratio at most $e$.

In the stochastic knapsack-cover instance let $p_\ell = \Pr[R = \ell]$ for $\ell \in \{0, 1, \cdots k\}$. Then it is clear that the adaptive optimum is $AD = c_n + \sum_{\ell=0}^{k} p_{k-\ell} \cdot OPT(\ell)$ where $OPT(\ell)$ is the optimal $\ell$-MST value. Note also that the sequence $\sigma$ corresponds to a randomized non-adaptive solution: item $n$ followed by the remaining $n-1$ items in the order that they first appear in $\sigma$. This has expected value $NA = c_n + \sum_{\ell=0}^{k} p_{k-\ell} \cdot \mathbb{E}_\sigma[\sigma(\ell)]$ where $\sigma(\ell)$ is the cost of the minimal prefix of $\sigma$ with reward at least $\ell$. So the adaptivity gap is at most:

$$\frac{NA}{AD} = \frac{c_n + \sum_{\ell=0}^{k} p_{k-\ell} \cdot \mathbb{E}_\sigma[\sigma(\ell)]}{c_n + \sum_{\ell=0}^{k} p_{k-\ell} \cdot OPT(\ell)} \le \max_\ell \frac{\mathbb{E}_\sigma[\sigma(\ell)]}{OPT(\ell)} \le e.$$

This also implies that the best deterministic non-adaptive policy costs at most $e \cdot AD$. ◀

## 5 Conclusion

The main (perhaps challenging) open question is to obtain a constant-factor approximation algorithm for Stochastic $k$-TSP in either the adaptive or non-adaptive setting. Another interesting question is the adaptivity gap, even in the special case of stochastic knapsack cover, where there is a wide gap between the lower bound of $e$ and upper bound of $O(\log^2 k)$.

## Acknowledgment

We thank Itai Ashlagi for initial discussions that lead to this problem definition.

## References

1 M. Adamczyk. Improved analysis of the greedy algorithm for stochastic matching. *Information Processing Letters*, 111(15):731–737, 2011.
2 M. Adamczyk, F. Grandoni, and J. Mukherjee. Improved approximation algorithms for stochastic matching. In *Proc. Annual European Symposium on Algorithms*, pages 1–12, 2015.
3 M. Adamczyk, M. Sviridenko, and J. Ward. Submodular stochastic probing on matroids. *Math. Oper. Res.*, 41(3):1022–1038, 2016.
4 Y. Azar, A. Madry, T. Moscibroda, D. Panigrahi, and A. Srinivasan. Maximum bipartite flow in networks with adaptive channel width. *Theoretical Computer Science*, 412(24):2577–2587, 2011.
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 N. Bansal and V. Nagarajan. On the adaptivity gap of stochastic orienteering. *Mathematical Programming*, 154(1-2):145–172, 2015.
7 A. Baveja, A. Chavan, A. Nikiforov, A. Srinivasan, and P. Xu. Improved bounds in stochastic matching and optimization. In *Proc. International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 124–134, 2015.
8 A. Bhalgat, A. Goel, and S. Khanna. Improved approximation results for stochastic knapsack problems. In *Proc. Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1647–1665. Society for Industrial and Applied Mathematics, 2011.
9 A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoff. Approximation algorithms for orienteering and discounted-reward TSP. *SIAM Journal of Computing*, 37(2):653–670, 2007.
10 K. Chaudhuri, B. Godfrey, S. Rao, and K. Talwar. Paths, trees, and minimum latency tours. In *Proc. Annual Symposium on Foundations of Computer Science*, pages 36–45, 2003.
11 C. Chekuri, N. Korula, and M. Pál. Improved algorithms for orienteering and related problems. *ACM Transactions on Algorithms*, 8(3):23, 2012.
12 N. Chen, N. Immorlica, A. R Karlin, M. Mahdian, and A. Rudra. Approximating matches made in heaven. In *Proc. International Colloquium on Automata, Languages and Programming*, pages 266–278. Springer, 2009.
13 M. Chrobak, C. Kenyon, J. Noga, and N. E. Young. Incremental medians via online bidding. *Algorithmica*, 50(4):455–478, 2008.
14 B. C Dean, M. X. Goemans, and J. Vondrák. Adaptivity and approximation for stochastic packing problems. In *Proc. Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 395–404. Society for Industrial and Applied Mathematics, 2005.

**15** B. C. Dean, M. X. Goemans, and J. Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.*, 33(4):945–964, 2008.

**16** A. Deshpande, L. Hellerstein, and D. Kletenik. Approximation algorithms for stochastic submodular set cover with applications to boolean function evaluation and min-knapsack. *ACM Transactions on Algorithms*, 12(3):42, 2016.

**17** J. Fakcharoenphol, C. Harrelson, and S. Rao. The $k$-traveling repairmen problem. *ACM Transactions on Algorithms*, 3(4), 2007.

**18** N. Garg. Saving an epsilon: a 2-approximation for the k-mst problem in graphs. In *Proc. ACM Symposium on the Theory of Computing*, pages 396–402, 2005.

**19** M. X. Goemans and Jan Vondrák. Stochastic covering and adaptivity. In *Proc. Latin American Symposium on Theoretical Informatics(LATIN)*, pages 532–543, 2006.

**20** D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

**21** N. Grammel, L. Hellerstein, D. Kletenik, and P. Lin. Scenario submodular cover. In *International Workshop on Approximation and Online Algorithms*, pages 116–128. Springer, 2016.

**22** S. Guha and K. Munagala. Multi-armed bandits with metric switching costs. *Automata, Languages and Programming*, pages 496–507, 2009.

**23** A. Gupta, R. Krishnaswamy, M. Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *Proc. Annual Symposium on Foundations of Computer Science*, pages 827–836, 2011.

**24** A. Gupta, R. Krishnaswamy, V. Nagarajan, and R. Ravi. Running errands in time: Approximation algorithms for stochastic orienteering. *Math. Oper. Res.*, 40(1):56–79, 2015.

**25** A. Gupta and V. Nagarajan. A stochastic probing problem with applications. In *Proc. Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 205–216, 2013.

**26** A. Gupta, V. Nagarajan, and S. Singla. Adaptivity gaps for stochastic probing: Submodular and XOS functions. In *Proc. Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1688–1702, 2017.

**27** Sungjin Im, Viswanath Nagarajan, and Ruben van der Zwaan. Minimum latency submodular cover. *ACM Trans. Algorithms*, 13(1):13:1–13:28, 2016.

**28** P. Kambadur, V. Nagarajan, and F. Navidi. Adaptive submodular ranking. In *Proc. Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 317–329, 2017.

**29** G. Lin, C. Nagarajan, R. Rajaraman, and D. P. Williamson. A general approach for incremental approximation and hierarchical clustering. *SIAM Journal of Computing*, 39(8):3633–3669, 2010.

**30** W. Ma. Improvements and generalizations of stochastic knapsack and multi-armed bandit approximation algorithms. In *Proc. Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1154–1163. Society for Industrial and Applied Mathematics, 2014.