1 Introduction

The *stochastic orienteering* problem, we are given a set V of n vertices in metric space (V, d). At each vertex $v \in V$ is a job with fixed reward r_v, which requires a stochastic processing time $S_v \sim \pi_v$ to claim. Starting from a fixed vertex $\rho \in V$, our goal is to create a policy for touring the vertices which maximizes expected reward, and such that the sum of traveling and processing times is no more than some budget B. This problem generalizes stochastic knapsack, a packing problem where we do not need to travel between jobs.

The main reference for our investigation is a paper on stochastic orienteering by Gupta et al. [GKNR12]. They give a $O(\log \log B)$-approximation for stochastic orienteering, and we will briefly review their method. In contrast to some of the other references, this result did not use a linear program relaxation (they mention that currently, no good one is known). The analysis gradually builds from several simplified versions of the problem.

2 Gupta et al. $O(\log \log B)$-approximation

Formally, we describe a policy as a decision tree $\mathcal{T}$ of the jobs to attempt, branching due the stochastic processing times of jobs. It is known that the optimal policy may have exponential size, and that stochastic orienteering is PSPACE-hard in general [DGV08]. For this reason, it is useful to distinguish between *adaptive* strategies and *non-adaptive* strategies as solutions. Non-adaptive strategies are simply an ordering of the vertices to attempt, decided before any stochastic values are determined. Adaptive strategies, in constrast, may change their decisions after observing random variable outcomes (these are what we traditionally consider for feasible solutions, and the optimal). By nature, non-adaptive strategies only require linear space to describe, but the *adaptivity gap* of stochastic orienteering is at least $O(\sqrt{\log \log B})$ [BN15]. That is, there are instances where no non-adaptive policy has reward better than $\Omega(1/\sqrt{\log \log B})$ the best adaptive policy. The solution constructed by Gupta et al. is a non-adaptive policy, so there is still a gap in this lower bound.

Gupta et al. use a series of simplifications (each leveraging the previous) to devise an approximation for stochastic orienteering.

1. The workhorse at the bottom is an approximation algorithm for "knapsack orienteering", which uses deterministic job rewards/sizes and splits the budget into two parameters $B = L + W$, where L may only be spent on traveling in the metric, and W may only be spent on processing jobs. This subproblem is solved using a reduction to deterministic orienteering, by *Lagrangian relaxation* on the job sizes (a technique from [GM09]).

2. Next, they show that this subproblem gives them an $O(1)$-approximation to the optimal non-adaptive policy $\mathrm{OPT}_{\mathrm{non}}$. The subproblem has deterministic parameters, so the authors perform a reduction from the stochastic case (but where the budget is still split as L and W). They use a technique called *truncated means* (used earlier by [DGV08] for stochastic knapsack) to essentially replace each stochastic job size with its expectation, while eliminating infeasible job realizations (i.e. $S_v > W$). For each $v \in V$, a deterministic size s_v is computed.

   $$\hat{s}_v = \min \{\mathbb{E}[S_v], W/2\}$$

   With the right discounting on rewards, the authors prove that some $W \in [0, B]$ provides a $O(1)$-approximation to the best non-adaptive policy through this reduction. Finally, they use an *exponential search* to find the optimal value of W. Specifically, the algorithm will attempt values of $W = B, B/2, B/4, \ldots, B/2^{\lfloor \log B \rfloor}, 0$. The "best" choice of W is always within a factor 2 of one of the attempted values, losing only a constant in approximation.

   Crucially, they prove a structure theorem showing that either a single vertex has reward $\Omega(\mathrm{OPT}_{\mathrm{non}})$, otherwise the knapsack orient subproblem solution for some W provides $\Omega(\mathrm{OPT}_{\mathrm{non}})$ reward. At a high level, they show that $\mathrm{OPT}_{\mathrm{non}}$ only has a small, constant probability of visiting/passing any vertex whose size is likely to overwhelm the remaining budget.

3. Surprisingly, the bound for adaptive policies uses the same algorithm. A similar structure theorem is proved: either a single vertex has $O(\log\log B)$ fraction of the optimal reward, otherwise some short adaptive policy has $O(\log\log B)$ fraction of the optimal reward. The difference is in the form of OPT, now a decision tree of possibly exponential size. To make the analysis simpler, they add dummy nodes to this tree so that each edge is a unit step in the metric.

   They truncate the paths of this tree along a frontier of nodes where the job size is likely to exceed the remaining budget. To obtain the same constant-probability bound on the probability of the stochastic process to pass this frontier, they use a Martingale analysis instead of the previous Chernoff bound. Averaging shows that a single path in the tree must therefore hold $\Omega(\text{OPT})$ expected reward, and therefore the choice of parameters proves that one of the knapsack orient instances has $\Omega(\text{OPT}/\log\log B)$ reward.

3 $O(\sqrt{\log\log B})$ adaptive gap

Bansal and Nagarajan [BN15] prove a lower bound on the adaptivity gap on a directed binary tree.

4 Constant Approximation of Stochastic Knapsack Orienteering

In [GKNR12], the $O(1)$-approximation algorithm for the deterministic knapsack orienteering, AlgKO, is used as a subroutine for the non-adaptive algorithm for StocOrient. AlgKO make uses of the technqiue *Lagrangian Relaxation*, which chooses some weakly coupled constraint and use the Lagrange multiplier as a threshold in designing the policy.

In the following analysis, we show how to extend this method to give an $O(1)$-approximation algorithm for the stochastic knapsack orienteering problem, by applying a technique similar to [GM09].

In the **stochastic knapsack orienteering** (StocKO), we are given two budgets, L which is the travel budget, and W which is the knapsack budget. There are n jobs, $v_1, v_2, \ldots, v_n$, each having reward r_{v_i}, and a random size S_{v_i}. Constrained on the total travelled distance being at most L and the total size of jobs being at most W, we want to maximize the amount of total reward obtained.

The main theorem is

Theorem 1. *There exists an algorithm that gives an $(8+\epsilon)$-approximation to the optimal stochastic knapsack orienteering problem.*

A non-adaptive policy can be represented as a path $P = \{v_1, v_2, \ldots, v_k\}$. $\forall v_i \in V$, $x_i = 1$ if v_i is on the optimal path. When executing the policy, The objective function is

$$\max \mathbf{E}\left[\mathsf{StocKO}(\mathbf{x})\right] = \max_P \sum_{i=1}^{k} r_{v_i} x_{v_i} \Pr\left[\sum_{j=1}^{i} S_{v_j} < B\right] \le \max_P \sum_{v\in V} r_v x_v = \mathsf{KO\text{-}OPT} \tag{1}$$

Where KO-OPT is the optimal solution to the deterministic knapsack orienteering problem (KO), whose size of job at vertex v is $\mathbf{E}[S_v]$. **Our goal is to find an algorithm whose expected reward is at a least $\frac{1}{\beta}$KO-OPT for some constant β.**

In the corresponding deterministic knapsack orienteering problem, we now have the travel constraint 2 and the knapsack constraint 3,

$$\sum_{(x_i,x_{i+1})\in P} d(x_i, x_j) \le L \tag{2}$$

$$\sum_{v\in V} \mathbf{E}[S_v]\, x_v \le W \tag{3}$$

4.1 Lagrangian Relaxation

The algorithm first find and approximation for the deterministic knapsack orienteering problem. We observe that the knapsack constraint 3 is a simple weakly coupled constraint. Therefore we use the Lagrange multipler:

$$\mathcal{L}(\mathbf{x}) = \max_{P} \sum_{v \in V} r_v x_v - \lambda \mathbf{E}[S_v] x_v = \max_{P} \sum_{v \in V} (r_v - \lambda \mathbf{E}[S_v]) x_v \tag{4}$$

Subject only to the travel constraint 2. There exists a λ such that $\mathcal{L}(x) = \text{KO-OPT}$. We now have a new **deterministic orienteering** (Orient) instance, whose job locations and travel constraint are the same as the original StocKO instance, but there is no longer the knapsack constraint, and each job v has reward $r_v - \lambda \mathbf{E}[S_v]$. Note that the optimal to this Orient instance satisfies

$$\textsf{Orient-OPT}(\lambda) = \sum_{v \in P} r_v - \lambda \mathbf{E}[S_v] \geq \textsf{KO-OPT} - \lambda W \tag{5}$$

Using results from [CKP12], we can obtain a constant approximation, $\textsf{Approx-Orient}(\lambda) \geq \frac{1}{\alpha}\textsf{Orient-OPT}(\lambda)$, where $\alpha = 2 + \epsilon$, $\forall \epsilon > 0$. We want to choose λ such that $\hat{R}_{\textsf{orient}}$ is not too small compared to KO-OPT. Find

$$\lambda^* = \max \left\{ \lambda \middle| \textsf{Approx-Orient}(\lambda) \geq \frac{\lambda W}{\alpha} \right\} \tag{6}$$

In particular, since $\lambda = \frac{\textsf{KO-OPT}}{2W}$ satisfies $\textsf{Approx-Orient}(\lambda) \geq \frac{\lambda W}{\alpha} = \frac{\textsf{KO-OPT}}{2\alpha}$, we know $\lambda^* \geq \frac{\textsf{KO-OPT}}{2\alpha}$, for which $\textsf{Approx-Orient}(\lambda^*)$ is a 2α-approximation of KO-OPT.

4.2 Convert the Solution

Let the reward obtained by following the path σ be $r(\sigma)$. To convert the approximate optimal path of Orient instance σ to a valid non-adaptive path of StocKO with large reward, we first observe the following fact, which is a variation of Observation 8 in [GKNR12]:

Fact 2. *If no single-vertex tour has reward $\Omega(OPT)$, we can skip all visits to "bad" jobs v, which are jobs such that $\Pr_{v \in \sigma}[S_v > W] \geq \frac{1}{2}$.*

The proof is based on the idea that after visiting each bad job, the probability that the knapsack constraint is not violated decreases geometrically by a factor of $\frac{1}{2}$. Hence the total expected reward from bad vertices does not exceed twice the expected reward obtained from the first bad vertex.

Therefore, we can deal with the single-vertex tours separately, and remove them in the following analysis. Our algorithm partitions σ greedily into c disjoint subpaths, $\sigma_1, \ldots, \sigma_c$, such that either σ_j contains a single job v such that $\mathbf{E}[v] \geq kW$, or $\sum_{v \in \sigma_j} \mathbf{E}[S_v] \leq kW$, $\forall j \in [1:c]$. Here k is a constant. Let y be a positive number such that $\sum_{v \in \sigma} \mathbf{E}[S_v] = yW$, then $c = \max\{1, \lfloor 2y/k \rfloor\}$. The algorithm finds the sub-path σ_{j^*} that has the largest reward, we show that this is only a constant factor smaller than KO-OPT.

$$\begin{aligned}
r(\sigma_{j^*}) &\geq \frac{r(\sigma)}{c} && (7)\\
&\geq \frac{\lambda^* y W + \lambda^* W/\alpha}{c} && (8)\\
&= \lambda^* W \left(\frac{y + 1/\alpha}{c} \right) && (9)\\
&\geq \lambda^* W \min\{\frac{y}{k} + \frac{1}{\alpha}, \frac{1}{2} + \frac{k}{2\alpha y}\} && \geq \frac{\lambda^* W}{\alpha} \geq \frac{\textsf{KO-OPT}}{2\alpha} \quad (10)\\
&\geq \frac{\textsf{StocKO-OPT}}{2\alpha} && (11)
\end{aligned}$$

It remains to show that if we use the policy of following σ_{j^*} in StocKO, we are still getting comparable amount of reward. By construction, one of the following two cases about σ_{j^*} is true:

1. σ_{j^*} contains a single vertex v such that $\mathbf{E}[S_v] \geq kW$. Since we know that $\Pr[S_v > W] < \frac{1}{2}$, if we only visit this vertex, with probability $\frac{1}{2}$, we are able to get its reward, which is at least $\frac{\textsf{StocKO-OPT}}{2\alpha}$ Therefore the expected reward of following σ_{j^*} in StocKO is at least $\frac{\textsf{StocKO-OPT}}{4\alpha}$.

2. σ_{j^*} contains a sequence of vertices such that $\sum_{v\in\sigma_{j^*}} \mathbf{E}[S_v] \leq kW$. By Markov's Inequality,

$$\Pr\left[\sum_{v\in\sigma_{j^*}} \mathbf{E}[S_v] \leq W\right] \geq 1-k \tag{12}$$

   With probability at least $1-k$, we can finish visiting the entire sub-path σ_{j^*}. Therefore the expected reward of following σ_{j^*} in StocKO is at least $(1-k)\frac{\textsf{StocKO-OPT}}{2\alpha}$.

If we choose $k = \frac{1}{2}$, we obtain the theorem 1.

5 Stochastic Orienteering with Cancellation

[GKNR12] also suggested briefly an idea to solve another extension of StocOrient that allows cancellation, for which we will provide a more detailed analysis here.

In this version, a strategy is allowed to abort the job while waiting for the job to finish. If it chooses to do so, it obtain a reward of 0 for this job, and it is not allowed to return to the job again. We cannot use the algorithm for StocOrient directly, since [GKMR11] has shown that even for the stochastic knapsack problem (the degenerate StocOrient where all jobs are located at the same location), allowing cancellation may improve the expected reward by a super-constant factor.

However, we can imagine that for each job located at a vertex v, we create up to $t_s = B - d(\rho, v)$ jobs colocated at v, where B is the budget and $d(\rho, v)$ is the distance between the starting vertex ρ and the vertex v. Each job $\langle s, t\rangle$, where $t \in [1 : t_s]$, corresponds to the policy of waiting for job s for up to t time, and abort the job if it takes more than t time. We therefore have a StocOrient instance with up to $O(nB)$ jobs.

However there are 2 issues that need to be addressed:

1. After the reduction, the number of jobs depends linearly on B, which could be large.

2. The optimal StocOrient could choose jobs $\langle s, t\rangle$ with different values of t. This is not feasible for the original problem of stochastic orienteering with cancellation.

Let OPT be the optimal expected reward of the original stochastic orienteering with cancellation problem. Due to issue 2, the optimal expected reward of the new StocOrient instance is at least OPT . To solve the two problem, we only pick $\langle s, t\rangle$ for t in a subset $I_s \subset [1 : t_s]$, which satisfies the following properties: $\forall t \in [1 : t_s]$ and $\forall s$,

$$\frac{f(s,t)}{2} \leq \max_{l\in I: l\leq t} f(s,l), \tag{13}$$

and

$$\sum_{l\in I: l\leq t} f(s,l) \leq 2f(s,t). \tag{14}$$

where $f(v,t)$ is the total expected reward obtained from $\langle s, t\rangle$, which is non-decreasing.

These conditions are similar to those in claim 14 of [GKNR12], and can be easily constructed by picking the largest t_i such that $f(s,t_i) \leq 2f(s,t_{i-1})$ at step i. It is easy to see that $|I_s| = O(\log B)$. If we only construct jobs $\langle s, t\rangle$ for $t \in I_s$, issue 1 is resolved. We will then show that even if the approximately optimal strategy for the new StocOrient instance picks multiple jobs $\langle s, t\rangle$ where $t \in I_s$, we can pick the job with largest expected reward and don't loose much.

Suppose the optimal policy for the original problem picks $\langle s, t^* \rangle$ as the policy for job s, then by 13, $\exists\ \hat{t} \in I$, *s.t.*

$$f(s, \hat{t}) \leq f(s, t^*) \leq 2f(s, \hat{t}) \tag{15}$$

Therefore the solution for the new StocOrient instance if we only construct jobs $\langle s, t \rangle$ for $t \in I_s$ is at least $\frac{\mathsf{OPT}}{2}$. We compute the approximate solution for this StocOrient instance, which is at least $\frac{\mathsf{OPT}}{2\alpha}$, where α is the approximation ratio for the StocOrient problem. The solution could contain multiple instances related to s: $\langle s, t_1 \rangle, \langle s, t_2 \rangle, \ldots, \langle s, t_k \rangle$, which makes the solution infeasible for the original problem. However, suppose $t_1 \leq t_2 \leq \ldots, \leq t_k$, if we only pick $\langle s, t_k \rangle$ and discard the rest for job s, by 14,

$$\sum_{i=1}^{k} f(s, t_i) \leq \sum_{l \in I : l \leq t_k} f(s, l) \leq 2f(s, t_k) \tag{16}$$

we only loose a factor of 2. And the total expected reward is at least $\frac{\mathsf{OPT}}{4\alpha}$

Therefore the algorithm is to construct jobs $\langle s, t \rangle$ for $t \in I_s$ for every job s, run the α-approximation algorithm for StocOrient for the new set of jobs, which returns a path representing a non-adaptive strategy. Within this path, for each job s that appears, only keep $\langle s, t \rangle$ with the largest time t. This strategy is a 4α-approximation of the optimal solution to the stochastic orienteering problem with cancellation.

The factor of 2 in 13 and 14 can be replaced by smaller constant β, however, this will increase the size of each I_s by a factor of $\frac{1}{\beta}$, which increases the runtime.

References

[BN15] Nikhil Bansal and Viswanath Nagarajan. On the adaptivity gap of stochastic orienteering. *Mathematical Programming*, 154(1-2):145–172, 2015.

[CKP12] Chandra Chekuri, Nitish Korula, and Martin Pál. Improved algorithms for orienteering and related problems. *ACM Transactions on Algorithms (TALG)*, 8(3):23, 2012.

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

[GKMR11] Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 827–836. IEEE, 2011.

[GKNR12] Anupam Gupta, Ravishankar Krishnaswamy, Viswanath Nagarajan, and R Ravi. Approximation algorithms for stochastic orienteering. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1522–1538. SIAM, 2012.

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