Constant Approximation for Stochastic Orienteering Problem with $(1+\epsilon)$-Budget Relaxiation

Yiyao Jiang(✉)

Institute for Interdisciplinary Information Sciences,
Tsinghua University, Beijing, China
jiangyy13@mails.tsinghua.edu.cn

Abstract. In the Stochastic Orienteering Problem (SOP), we are given finite metric space (V, d) and a starting point $\rho \in V$. Each $v \in V$ has an associated reward $r_v \geq 0$ and random completion time S_v, where the distribution of each S_v is know (once a reward has been earned, it cannot be earned again); the time cost of traveling from $u \in V$ to $v \in V$ is $d(u, v)$. The goal is to sequentially visit vertices and complete tasks in order to maximize the total rewards within 1 unit of time (after normalization). In this paper, we present a nonadaptive $O(\epsilon^{-14})$-approximation for (the original, adaptive) SOP when we relax the unit time budget to $(1+\epsilon)$, $0 < \epsilon < 1$.

1 Introduction

In the competitive practice of orienteering, participants are given a map with marked locations. Starting from a given point, they must try to visit as many locations as possible within a given time limit. Each point has a given reward, and the goal for competitors is to maximize the total reward within the time limit. This problem was first studied in [8] from an algorithmic perspective; see [13] for a more recent survey.

A similar problem to Orienteering Problem (OP) is the Prize Collecting Traveling Salesman Problem (PCTSP) [5]. In the latter, the salesman's costs are the time spent traveling between points, plus a point-dependent penalty that is incurred for each point that he fails to visit. PCTSP's goal is to minimize the traveling time plus the penalties of the points. Comparing with OP, PCTSP is not limited in time, whereas OP is strictly bounded with time limit.

1.1 The (Simple) Orienteering Problem

An instance of the Orienteering Problem (OP) consists of a metric space (V, d) with $|V| = n$ points and distances $d(u, v) \in \mathbb{R}^+$ for each $(u, v) \in V \times V$; moreover, the instance specifies a starting point $\rho \in V$, a total time budget B, and a reward $r_v \in \mathbb{R}^+$ for each $v \in V$. The object is to chart a path that maximizes

Y. Cao and J. Chen (Eds.): COCOON 2017, LNCS 10392, pp. 297–308, 2017.
DOI: 10.1007/978-3-319-62389-4_25

the sum of rewards r_v while staying within the time budget B, where the distances $d(u,v)$ are interpreted as time costs. In this setting, and as all quantities are real numbers, we can assume without loss of generality that $B = 1$ after normalization.

In 1998 Arkin et al. [1] obtained a 2-approximation for OP with planar Euclidean distances. Furthermore, Chen and Har-Peled [7] obtained a polynomial-time approximation scheme (PTAS) for OP in fixed-dimensional Euclidean space.

For general metric spaces (V, d), Blum et al. [2] obtained the first constant-factor approximation (with factor 4) in 2007. The best result until now is a $(2+\epsilon)$-approximation algorithm by Chekuri et al. [6].

The Orienteering Problem finds many natural applications. For example, a traveling salesman with limited time [12] (who must maximize his total profit in one day).

1.2 Stochastic Orienteering Problem

In the Stochastic Orienteering Problem (SOP) [9] a job is associated to each vertex $v \in V$, which must be completed at v in order to obtain the reward r_v. The time necessary to complete the job, moreover, is randomly distributed. For example, a salesman may need to wait for his customer to come downstairs, or a tourist need time to enjoy the scenery, and so on.

Generally, one is not allowed to give up a job until it is finished. In certain models one is allowed to give up a job and its reward after starting the job, while being forbidden from re-trying to complete the job later, once it has been given up.

The time required by job v is a random variable S_v, where S_v has a known discrete distribution π_v for each $v \in V$, where π_v modeled as a function from $\mathbb{R}^+$ into $[0, 1]$; the distributions $\{\pi_v : v \in V\}$ are thus part of an instance of SOP.

We note that SOP can be considered as a kind of combination of OP and of the Stochastic Knapsack Problem (SKP) [11]. The Stochastic Knapsack Problem, indeed, corresponds to an instance of SOP in which $d(u,v) = 0$ for all $(u,v) \in V \times V$. For SKP, moreover, a $(2+\epsilon)$-approximation algorithm is known [4].

For a non-adaptive variant of SOP in which the path is chosen in advance and in which the distances $d(u,v)$ are restricted to integers, Gupta et al. [9] have shown a constant factor approximation algorithm. In that setting (and due to the integer restriction on distances) the time budget B is not normalized to 1, and remains a parameter of the problem. In fact, Gupta et al. show that their algorithm is an $O(\log\log B)$-approximation algorithm for SOP with arbitrary *adaptive* policies. In the same formal setting, moreover, Bansal and Nagarajan [3] have shown that a $\Omega((\log\log B)^{1/2})$ gap between adaptive and non-adaptive policies is unavoidable. This contrasts with the case of SKP, for which non-adaptive policies can approximate adaptive policies to within a constant factor [4] (In OP the distinction between adaptive and non-adaptive policies is moot.).

In this paper we present an $O(\epsilon^{-14})$-approximation for the (original, adaptive) SOP with time budget $(1+\epsilon)B = 1+\epsilon$, $0 < \epsilon < 1$. More precisely, we show a *non-adaptive* policy of time budget $1+\epsilon$ such that the expected reward of the optimal *adaptive* policy at time budget 1 is at most $O(\epsilon^{-14})$ times the expected reward of our [non-adaptive, time $1+\epsilon$] policy. Our algorithm therefore achieves an $O(1)$-approximation for any constant $0<\epsilon<1$.

Theorem 1. *There is a polynomial time, non-adaptive $O(\epsilon^{-14})$-approximation for SOP with respect to a relaxed time budget of $(1+\epsilon)B$, $0<\epsilon<1$.*

2 Definitions and Notations

We write $\mathbb{R}^+$ for the nonnegative real numbers $[0,\infty)$. In this paper all quantities are real-valued.

2.1 Orienteering

As introduced in Sect. 1, an instance of the Orienteering Problem (OP) is defined on an underlying metric space (V,d) with $|V|=n$ points and distances $d(u,v)\in\mathbb{R}^+$ for each $(u,v)\in V\times V$. We are given a starting "root" point $\rho\in V$ and a total time budget B. An instance of OP thus corresponds to a tuple $I_{\text{O}}=(V,d,\{r_v\}_{v\in V},B,\rho)$.

2.2 Stochastic Orienteering

In the Stochastic Orienteering Problem (SOP) each point $v\in V$ is related to a unique stochastic job. The job associated to v has a fixed reward $r_v\geq 0$, and a random variable of processing time size S_v with a known but arbitrary discrete probability distribution $\pi_v: U\to[0,1]$, where $U\subseteq\mathbb{R}^+$ is a countable set, such that $\sum_{t\in U}\pi_v(t)=1$ for all $v\in V$. (Time size is the time that we have to wait at the point before receiving the reward for job v. Note that we can take the same underlying time set U for all $v\in V$, wlog.) An instance of SOP thus corresponds to a tuple $I_{\text{SO}}=(\pi,V,d,\{r_v\}_{v\in V},\{S_v\}_{v\in V},B,\rho)$.

At each step we travel to a point v from a current point u (at time cost $d(u,v)$) and then process the job v, at time cost S_v, and once a job v is selected, one must wait for v to complete. If we are still within the time budget B when the job is completed we receive the reward r_v, and choose a new point as the next destination.

For generality, $\forall t\in U$ we assume $t\in[0,B]$. If one time size $t>B$, we can just truncated it to B, since we will always stop at time budget B before completing the job and never get the reward. Also, we can assume wlog that $d(u,\rho)\leq B$ for all $u\in V$, which also implies that $d(u,v)\in[0,2B]$ for all u, $v\in V$.

2.3 Task Orienteering Problem

Our analysis refers to a simplified version of the Stochastic Orienteering Problem known as the "Task Orienteering Problem" (TOP); TOP corresponds to an instance of SOP in which each time size S_v is deterministic. An instance of TOP thus corresponds to a tuple $I_{\text{TO}}=(V,d,\{r_v\}_{v\in V},\{s_v\}_{v\in V},B,\rho)$ with $s_v\in\mathbb{R}^+$ for each $v\in V$.

3 Algorithm

We begin with an analysis of the Task Orienteering Problem described in Sect. 2.3, to be used as a component in our main analysis.

3.1 An Algorithm for TOP

We first provide a polynomial time algorithm AlgTO (Algorithm 1) and then prove that AlgTO gives an $O(1)$-approximation to TOP.

Definition 1 *(Valid* OP *Instance).* *Given an instance $I_{\mathrm{TO}} = (V, d, \{r_v\}_{v\in V}, \{s_v\}_{v\in V}, B, \rho)$ of TOP, we define the following valid OP instance $I_{\mathrm{O}}(I_{\mathrm{TO}}) := (V', d', \{r_v\}_{v\in V'}, B, \rho)$, where*

1. $V' := \{v \in V : d(\rho, v) + s_v \le B\}$;
2. $d'(u, v) := d(u, v) + s_v/2 + s_u/2$ for all $u, v \in V'$, $u \neq v$;
3. $d'(u, u) := 0$ for all $u \in V$;
4. $r'_v := r_v$ for all $v \in V'$.

Lemma 1. *The function $d' : V' \times V' \to \mathbb{R}^+$ constructed in Definition 1 is a metric.*

The proof is easy.

Algorithm 1. AlgTO for Task Orienteering on input $I_{\mathrm{TO}} = (V, d, \{r_v\}_{v\in V}, \{s_v\}_{v\in V}, B, \rho)$

```
1. let $I_O(I_{\mathrm{TO}}) := (V', d', \{r'_v\}_{v\in V'}, B, \rho)$;
2. run AlgOrient on the (Simple) Orienteering Problem $I_O$,
3.    get the path $P$, the reward $\mathrm{OPT}'_{\mathrm{O}} = \sum_{v\in P} r'_v$, and the ending point $\rho'$;
4. compare $\sum_{v\in P\setminus\{\rho'\}} r'_v$ and $r'_{\rho'}$;
5. output the path
```

$$P' := \begin{cases} \rho \longrightarrow \rho', & \text{if } r'_{\rho'} \ge \sum_{v\in P\setminus\{\rho'\}} r'_v, \\ P\setminus\{\rho'\}, & \text{otherwise;} \end{cases}$$

```
6.    and the reward $\mathrm{OPT}'_{\mathrm{TO}} := \sum_{v\in P'} r'_v$.
```

Theorem 2. *Algorithm* AlgTO *(Algorithm 1) is a polynomial time $O(1)$-approximation for the Task Orienteering Problem.*

Proof. Assume the optimal result for Task Orienteering problem I_{TO} is $\mathrm{OPT}_{\mathrm{TO}}$. Now we prove that $\mathrm{OPT}'_{\mathrm{TO}}$ returning by the polynomial time algorithm AlgTO (Algorithm 1) is an $\Omega(1)$-approximation to $\mathrm{OPT}_{\mathrm{TO}}$.

In Algorithm 1 line 1, as in Definition 1 item 1, we delete all points $v \in V$ which are obviously not in the optimal path of I_{TO}, because even we only reach to one point v from the starting point ρ, we still do not have enough time to reach the point v and complete the job with time $d(\rho, v) + s_v > B$.

In Definition 1 item 2, for each point $v \in V'$, we divide the fixed job time s_v by 2, and add $s_v/2$ to the lengths of all edges adjacent to point v. (Note that, although we add $s_v/2$ to the lengths of all edges adjacent to point v, in real algorithm path, we only count twice of $s_v/2$ as doing the job in time s_v: the edge into point v, and the edge out of point v). Define the new distance as $d'(u, v)$.

In Lemma 1, we prove that (V', d') is a metric, thus we create an Orienteering Problem instance $I_{\mathrm{O}} = (V', d', \{r_v\}_{v\in V}, B, \rho)$ successfully in Algorithm 1 step 1.

In Algorithm 1 line 2, we use the polynomial time algorithm AlgOrient (by Blum et al. [2] mentioned in Sect. 2.1) on OP instance I_{O}. And in line 3, the path P of reward $\mathrm{OPT}'_{\mathrm{O}}$ is a constant approximation of the optimal solution for I_{O} by algorithm AlgOrient. Note that any path in I_{TO} including the optimal path of I_{TO} is surely a feasible path in I_{O} (since the only change is the distances in I_{TO} are longer), so $\mathrm{OPT}'_{\mathrm{O}} = \Omega(\mathrm{OPT}_{\mathrm{TO}})$.

Note that in path P, we always get in and out of a point v except for the starting point ρ and ending point ρ'. So the $s_v/2$ are always counted twice, and in I_{TO} we do complete all the jobs on the path except for ρ and ρ'. Thus the only incidence we cannot complete path P in I_{TO} is the jobs on ρ and ρ' in path P.

At the starting point ρ in I_{TO}, we can always create a fake "starting point" $\rho^\star$ which is the same $d'(,)$ as ρ but $d'(\rho, \rho^\star) = d'(\rho^\star, \rho) = d'(\rho^\star, \rho^\star) = 0$, $s_{\rho^\star} = 0$ and $r_{\rho^\star} = 0$, and we set ρ as a normal point. Run AlgTO (Algorithm 1) on I_{TO} with $\rho^\star$. If ρ is still in path, we just start from ρ and use s_ρ time to get the reward r_ρ then go through path P; else, ρ is not in path, we also start from ρ, but give up the job and reward on ρ, then go through path P. All these changes do not change the $\mathrm{OPT}'_{\mathrm{O}}$, and do not effect on $\mathrm{OPT}_{\mathrm{TO}}$. Thus, we now only need to concern about ρ'.

In Algorithm 1 line 4, we compare the reward $r'_{\rho'}$ at the ending point ρ' (note that $d(\rho, \rho') + s_{\rho'} \leq B$ in Definition 1 indicates that path $\rho \longrightarrow \rho'$ is feasible in I_{TO}), and the rewards $\sum_{v\in P\setminus\{\rho'\}} r'_v$ of all the points in path $\widehat{P}$ except for ρ'. Since $\mathrm{OPT}'_{\mathrm{O}} = \sum_{v\in P} r'_v = r'_{\rho'} + \sum_{v\in P\setminus\{\rho'\}} r'_v$, $\max\{r'_{\rho'}, \sum_{v\in P\setminus\{\rho'\}} r'_v\} \geq \mathrm{OPT}'_{\mathrm{O}}/2 = \Omega(\mathrm{OPT}_{\mathrm{TO}})$.

In Algorithm 1 line 5 and 6, we thus choose the larger one between $r'_{\rho'}$ and $\sum_{v\in P\setminus\{\rho'\}} r'_v$ as $\mathrm{OPT}'_{\mathrm{TO}}$. If $r_{\rho'}$ is larger, then we just go to a single point ρ' and complete the job; if the other is larger, we just ignore ρ', go and do the jobs through the path P except for ρ'.

Thus, we have a polynomial time $O(1)$-approximation algorithm AlgTO for Task Orienteering problem.

3.2 The Algorithm for Stochastic Orienteering

Definition 2 *(Valid* TOP *Instance). Given an instance* $I_{\mathrm{SO}} = (\pi, V, d, \{r_v\}_{v\in V}, \{S_v\}_{v\in V}, B, \rho)$ *of SOP, and a value* $\epsilon > 0$*, we define the following valid TOP* *instance with parameter* ϵ *to be* $I_{\text{TO}}(\epsilon, I_{\text{SO}}) := (\widehat{V}, \widehat{d}, \{\widehat{r}_u\}_{\forall u \in \widehat{V}}, \{\widehat{s}_u\}_{\forall u \in \widehat{V}}, \widehat{B}, \widehat{\rho})$, *where we*

1. *define* $\widehat{V}$*: let* $\widehat{V} := V$,
   for all $u \in V$*, if* $d(\rho, u) > B$, $\widehat{V} := \widehat{V} \backslash \{u\}$,
   create a virtual point $\widehat{\rho}$, $\widehat{V} := V \cup \{\widehat{\rho}\}$*;*
2. *for all* $u, v \in \widehat{V}$*, define distances* $\widehat{d}(u, v)$*: if* $u, v \in V$, $\widehat{d}(u, v) := d(u, v)$,
   for all $u \in V$*, set* $\widehat{d}(u, \widehat{\rho}) = \widehat{d}(\widehat{\rho}, u) = B$,
   set $d(\widehat{\rho}, \widehat{\rho}) = 0$*;*
3. *for all* $u \in \widehat{V}$*, define rewards* $\widehat{r}_u$*: if* $u \in V$, $\widehat{r}_u = r_u$*, set* $\widehat{r}_{\widehat{\rho}} = 0$*;*
4. *for all* $u \in \widehat{V}$*, define deterministic job times* $\widehat{s}_u$*: if* $u \in V$, $\widehat{s}_u = E[S'_u]$,
   where $S'_u = \min\{S_u, B\}$,
   set $\widehat{s}_{\widehat{\rho}} = 0$*;*
5. *define time bound* $\widehat{B}$*: let* $\widehat{B} := (1 + \epsilon^{13})B$*;*
6. *define the starting point: set* $\widehat{\rho}$ *to be the new starting point.*

Lemma 2. *The* $(\widehat{V}, \widehat{d})$ *of the TOP instance* I_{TO} *in Definition* 2 *is a metric space.*

The proof is easy.

Algorithm 2. Algorithm AlgSO for SOP on input $I_{SO} = (\pi, V, d, \{r_v\}_{v \in V}, \{S_v\}_{v \in V}, B, \rho)$ and parameter $0 < \epsilon < 1$

```
1. for all $v \in V$ do
2.     let $R_v := r_v \cdot \Pr_{S_v \sim \pi_v}[S_v \leq (B - d(\rho, v))]$ be the expected reward of the single-node tour from $\rho$ to $v$;
3. w.p. 1/2, just visit the point $\widehat{v}$ with the highest $R_{\widehat{v}}$ and exit.
4. let $I_{\text{TO}}(\epsilon, I_{\text{SO}}) := (\widehat{V}, \widehat{d}, \{\widehat{r}_u\}_{\forall u \in \widehat{V}}, \{\widehat{s}_u\}_{\forall u \in \widehat{V}}, \widehat{B}, \widehat{\rho})$;
5. run AlgTO (Algorithm 1) on the valid TOP instance $I_{\text{TO}}(\epsilon, I_{\text{SO}})$,
6.     get the path $P$, the reward $\widehat{\text{OPT}_{\text{TO}}}$;
7. replace the starting point $\widehat{\rho}$ in path $P$ of $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ by $\rho$ and output path $\widehat{P}$;
8. go through path $\widehat{P}$ in $I_{\text{SO}}$ with time budget $(1 + \epsilon)B$ and output the total reward $\widehat{\text{OPT}_{\text{SO}}}$.
```

We want to prove that Algorithm 2 is a polynomial time $O(\epsilon^{-14})$-approximation for Stochastic Orienteering problem with respect to a relaxed time budget of $(1 + \epsilon)B$, $0 < \epsilon < 1$.

Define the expected reward on optimal adaptive policy on I_{SO} as OPT, and define the expected reward on optimal policy on $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ as OPT_{TO}.

We design to have either a single-point tour $\rho \longrightarrow v$ of expected reward $R_v = \Omega(\text{OPT})$ with 50 % chance, or the path $\widehat{P}$ on I_{SO} of reward $\widehat{\text{OPT}_{\text{SO}}} = \Omega(\text{OPT})$ with 50 % chance. For this purpose, we want the following theorem.

Theorem 3. *Given an instance I_{SO} for which an optimal adaptive strategy has an expected reward of* OPT*, either there is a single-point tour with expected reward $\Omega(\text{OPT})$, or the valid Task Orienteering instance $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ has reward $\text{OPT}_{\text{TO}} = \Omega(\text{OPT})$, with constant approximation parameter ϵ^{14}.*

We will prove Theorem 3 later in Sect. 4. For now, let us assume that Theorem 3 is correct, and use it to complete the proof of Theorem 1. The following proof is for Theorem 1.

3.3 Proof for Theorem 1

Suppose we enter Algorithm 2 line 4, and in line 6, AlgTO (Algorithm 1) finds a path $P = (\widehat{\rho}, v_1, v_2, \ldots, v_k)$ with reward $\widehat{\text{OPT}_{\text{TO}}}$. And then we get the path $\widehat{P} = (\rho, v_1, v_2, \ldots, v_k)$.

Lemma 3. *In $I_{\text{SO}} = (\pi, V, d, \{r_v\}_{v \in V}, \{S_v\}_{v \in V}, B, \rho)$, for any point $v_i \in \widehat{P}$, the probability of successfully reaching to v and finishing the job at v before violating the budget $(1+\epsilon)B$ is at least $1 - \epsilon^{13}$.*

Proof. Note that now we relax the time budget on I_{SO} to $(1+\epsilon)B$, but we still have time budget $(1+\epsilon^{13})B$ on $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ as in Definition 2.

In Definition 2, for $I_{\text{TO}}(\epsilon, I_{\text{SO}})$, $\sum_{i=1}^{k}(\widehat{s}_{v_i} + \widehat{d}(v_{i-1}, v_i)) \leq (1+\epsilon^{13})B$, $S'_{v_i} = \min\{S_{v_i}, B\}$, and $\widehat{s}_{v_i} = E[S'_{v_i}]$. Since $\widehat{d}(\widehat{\rho}, v_1) = B$, $E[S'_{v_1}] + \sum_{i=2}^{k}(E[S'_{v_i}] + \widehat{d}(v_{i-1}, v_i)) \leq \epsilon^{13}B$.

Define $S^{\star}_{v_i} = \begin{cases} S'_{v_1} & \text{if } i = 1, \\ S'_{v_i} + \widehat{d}(v_{i-1}, v_i) & \text{if } 2 \leq i \leq k. \end{cases}$

Then $\sum_{i=1}^{k} E[S^{\star}_{v_i}] \leq \epsilon^{13}B$. By using Markov's inequality, for each $j \leq k$, we have

$$\Pr[\sum_{i=1}^{j} S^{\star}_{v_i} \leq \epsilon B] \geq 1 - \epsilon^{13}$$

The change from S_u to S'_u does not effect on optimal adaptive policy on I_{SO}, the distance $d(\rho, v_1) \leq B$, and $d(v_{i-1}, v_i) = \widehat{d}(v_{i-1}, v_i)$. So in path $\widehat{P}$ of I_{SO} with time budget $(1+\epsilon)B$, for each point v_j $(1 \leq j \leq k)$, the probability we successfully reach to point v_j, and complete the job on point v_j is not less than $\Pr[\sum_{i=1}^{j} S^{\star}_{v_i} \leq \epsilon B]$. Thus the probability of successfully reaching to v and finishing the job at v before violating the budget $(1+\epsilon)B$ is at least $1-\epsilon^{13}$. And this complete the proof for Lemma 3.

Proof (The proof of Theorem 1). We prove that AlgSO (Algorithm 2) is a polynomial time $O(\epsilon^{-14})$-approximation for Stochastic Orienteering problem with respect to a relaxed time budget of $(1+\epsilon)B$, $0 < \epsilon < 1$.

We assume that Theorem 3 is correct. Then either there is a single-point tour with expected reward $\Omega(\text{OPT})$, or the valid Task Orienteering instance $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ has reward $\text{OPT}_{\text{TO}} = \Omega(\text{OPT})$, with a constant approximation parameter ϵ^{14}.

1. There is a single-point tour $\rho \longrightarrow v$ satisfies $R_v = \Omega(\epsilon^{14}\text{OPT})$.
   We have 50% chance to obtain the highest $R_{\widehat{v}}$. And $R_{\widehat{v}} \geq R_v = \Omega(\epsilon^{14}\text{OPT})$.
2. The valid Task Orienteering instance $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ has reward $\text{OPT}_{\text{TO}} = \Omega(\text{OPT})$.
   We have 50% chance to get $\widehat{\text{OPT}}_{\text{SO}}$.
   In Algorithm 2 line 6, by applying Theorem 2, AlgTO (Algorithm 1) will find a path $P = (\widehat{\rho}, v_1, v_2, \ldots, v_k)$ with reward $\widehat{\text{OPT}}_{\text{TO}} = \Omega(\text{OPT}_{\text{TO}})$. By assuming Theorem 3 is correct, we have $\text{OPT}_{\text{TO}} = \Omega(\epsilon^{14}\text{OPT})$. Then $\widehat{\text{OPT}}_{\text{TO}} = \Omega(\text{OPT}_{\text{TO}}) = \Omega(\epsilon^{14}\text{OPT})$.
   Now applying Lemma 3 on path $\widehat{P}$ that we get in Algorithm 2 line 7, and that gives us an expected reward of at least $\widehat{\text{OPT}}_{\text{SO}} = \widehat{\text{OPT}}_{\text{TO}} \cdot (1 - \epsilon^{13}) = \Omega(\epsilon^{14}\text{OPT})$.

So we always have at least 50% probability to get $\Omega(\epsilon^{14}\text{OPT})$ by Algorithm 2, and thus we complete the proof of Theorem 1.

4 Optimal Policy for Stochastic Orienteering

The main method to deal with the huge optimal decision tree defined in Sect. 4.1 is Discretization and Block Policy [10]. Since all quantities are all real-valued, by scaling, we can assume $B = 1$ and the size of each item is distributed between 0 and B. The relaxed time budget $B + O(\varepsilon)$ should be less than $2B$.

4.1 Policy and Decision Tree

A policy σ for an SOP instance (π, V, d, B) can be represented as a decision tree $T_\sigma(\pi, V, d, B)$.

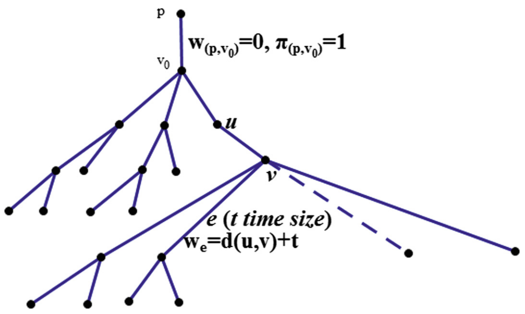

Fig. 1. Decision tree T_σ

In Fig. 1, each node of T_σ is labeled by a vertex $v \in V$, with the root of T_σ being labeled by ρ. The vertices on a path from the root to a node are distinct, hence T_σ has depth at most $n = |V|$. A node of T_σ has countably many children, or one for each element of U, where $U = \{t \in \mathbb{R}^+ : \pi_v(t) > 0 \text{ for some } v \in V\}$ is the (countable) support set of the time size distributions. Let $z \in T_\sigma$ be a node of label $v \in V$, and let e be the t-th edge emanating from z, $t \in U$; then e has *probability* $\pi_e := \pi_v(t) = \Pr[S_v = t]$; moreover if z's parent node is node $y \in T_\sigma$ of label $u \in V$, then e has *weight* $w_e := d(u, v) + t$.

We will sometimes refer to nodes in T_σ by their label. (With little chance of confusion.) For a node $v \in T_\sigma$, we define $P(v)$ to be the path of edges of leading from the root to v. We define $W(v) := \sum_{e\in P(v)} w_e$ and $\Phi(v) := \prod_{e\in P(v)} \pi_e$. Thus $W(v)$ is the total time elapsed before reaching v and $\Phi(v)$ is the probability of reaching v.

We note that a given policy σ will be "compatible" with any tuple (π', V', d', B') such that $V' = V$ and such that π' has the same support U as π. Namely, the modified π' and d' will induce modified edge weights w'_e and modified edge probabilities p'_e. We emphasize this by including (π, V, d, B) as arguments to T_σ (though we will always use the same V for a given policy σ).

We let

$$R(\sigma, \pi, V, d, B) = \sum_{v\in T_\sigma, e=(v,u): W(v)+w_e \le B} \pi_e \cdot r_v \Phi(v)$$

denote the expected reward that policy σ achieves with respect to the metric d, time size distributions π, and time budget B. We use **OPT** to denote the expected reward of the optimal adaptive policy.

For the following lemma, we also consider a modified node label $X_v := S_v + d(u, v)$, where u is the parent and v is the child. Thus, while $w_e = t + d(u, v)$ is a number, X_v is a random variable (Moreover, X_v "ignores" the fact that edge e is the t-th edge of v, i.e., that edge e is associated to a particular outcome of S_v.).

Lemma 4 (*part of Lemma 2.4 in* [4]). *For any policy σ on instance (π, V, d, B), there exists a policy σ' such that*

$$R(\sigma', \pi, V, d, B) = (1 - O(\epsilon))R(\sigma, \pi, V, d, B),$$

and for any realization path P in $T_{\sigma'}(\pi, V, d, B)$, $\sum_{v\in P} E[X_v] = O(B/\epsilon)$.

Here we mention that policy σ' is just cutting some branches on policy σ to ensure $\sum_{v\in P} E[X_v] = O(B/\epsilon)$. Note that policy σ is all designed by us, we surely may have a node v that $\sum_{v\in P} E[X_v] > O(B/\epsilon)$ though v has no contribution to reward.

4.2 Discretization

We present how to discretize the decision tree in Sect. 4.1 with given ϵ. We use the discretization method similar in paper [10] Sect. 2.1. We split all jobs into two parts: small size vertices and big size vertices, and discretize them separately.

Denote $\widetilde{S}_v$ for value of vertex v after discretization, and the new distribution $\widetilde{\pi}_v$. And for each node v in the tree, define $\widetilde{X}_v := \widetilde{S}_v + d(u, v)$ as a random variable which is similar in the definition of X_v in Sect. 4.1.

1. Small size region if $S_v \leq \epsilon^4$.
   There exists a value $0 \leq h \leq \epsilon^4$, such that $\Pr[S_v \geq h | S_v \leq \epsilon^4] \cdot \epsilon^4 = E[S_v | S_v \leq \epsilon^4]$. Then set: $\widetilde{S}_v = \begin{cases} 0, & 0 \leq S_v < h; \\ \epsilon^4, & h \leq S_v \leq \epsilon^4; \\ S_v, & S_v > \epsilon^4. \end{cases}$
2. Large size region if $S_v > \epsilon^4$.
   We simply discretize it as $\widetilde{S}_v = \lfloor \frac{S_v}{\epsilon^5} \rfloor \epsilon^5$.

We denote the set of the discretized size by $S = \{s_0, s_1, \cdots, s_{z-1}\}$ where $s_0 = 0$, $s_1 = \epsilon^5$, $s_2 = 2\epsilon^5, \ldots, s_{z-1}$. Note that $s_1 = \epsilon^5, s_2 = 2\epsilon^5, \ldots, s_{1/\epsilon-1} = \epsilon^4 - \epsilon^5$ are also in S though their probability is 0. The total number of values of time size is $|S| = z = O(B/\epsilon^5)$ which is a constant.

4.3 Canonical Policies

We need to use *canonical policies* introduced in [4]. A policy $\widetilde{\sigma}$ is a canonical policy if it makes decisions based on the discretized sized of vertices, but not their actual sizes. Under the canonical policy, we will keep trying new vertices if the total discretized size budget does not exceed, even the actual budget overflows. But no reward will get from these vertices which make actual budget exceeding.

Lemma 5 (*Lemma 4.2 in* [10]). *Let π be the distribution of vertice size 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*

$$R(\widetilde{\sigma}, \widetilde{\pi}, V, d, (1+4\epsilon)B) = (1 - O(\epsilon))R(\sigma, \pi, V, d, B);$$

2. *For any canonical policy $\widetilde{\sigma}$,*

$$R(\widetilde{\sigma}, \pi, V, d, (1+4\epsilon)B) = (1 - O(\epsilon))R(\widetilde{\sigma}, \widetilde{\pi}, V, d, B).$$

Proof sketch: for the first result, we prove that there is randomized canonical policy σ_r such that $R(\sigma_r, \widetilde{\pi}, V, \widetilde{d}, (1+4\epsilon)B) = (1 - O(\epsilon))R(\sigma, \pi, V, d, B)$. Thus such a deterministic policy $\widetilde{\sigma}$ exists. The randomized policy σ_r is derived from σ as follows. T_{σ_r} keeps the same tree structure as T_σ. If σ_r visits a vertex v and observes a discretized size $s \in S$, it chooses a random branch in σ among those sizes that are mapped to s. Let $t_1, t_2, \ldots, t_k$ are possible size realization of s as per size distribution π and let $\pi_v(t_1), \pi_v(t_2), \ldots, \pi_v(t_k)$ be the corresponding probabilities. Hence we choose one of the branches corresponding to size $t_1, t_2, \ldots, t_k$ w.p. $\pi_v(t_1), \pi_v(t_2), \ldots, \pi_v(t_k)$ (normalized) respectively. For more detail, see Lemma 4.2 in [10].

From Lemma 5, we conclude that the reward will not loss too much if we using canonical policies to applying the policies with $T_{\widetilde{\sigma}}$ described in Sect. 4.2, but not T.

4.4 Block Tree

Now we partition the decision tree $T_{\widetilde{\sigma}}$ created in Sect. 4.3 into blocks.

For any node v in the decision tree $T_{\widetilde{\sigma}}$, we define the *leftmost path of* v to be the realization path which starts at v,ends at a leaf, and consists of only edges corresponding to size zero. We define the *block* starting with node v (denote as seg(v)) in $T_{\widetilde{\sigma}}$ as the maximal prefix of the leftmost path of v such that: If $E[\widetilde{X}(v)] > \epsilon^{13}$, seg($v$) is the singleton node $\{v\}$. Otherwise, $E[\widetilde{X}(seg(v))] \le \epsilon^{13}$.

We partition $T_{\widetilde{\sigma}}$ into blocks as follows. We say a node v is a *starting node* if v is a root or v corresponds to a non-zero size realization of its parent. For each starting node v, we greedily partition the leftmost path of v into blocks, i.e., delete seg(v) and recurse on the remaining part.

Lemma 6. *A canonical policy* $\widetilde{\sigma}$ *with expected reward* $(1 - O(\epsilon))\mathrm{OPT}$ *can be partitioned into several blocks that satisfy the following properties:*

1. *There are at most* $O(\epsilon^{-14})$ *blocks on any root-leaf path in the decision tree.*
2. *There are* $|S| = O(B/\epsilon^5)$ *children for each block.*
3. *Each block* M *with more than one node satisfies that* $\sum_{b\in M} E[\widetilde{X_b}] \le \epsilon^{13}$.

Proof. 1. Fix a particular root-to-leaf path R. Let us bound the number of blocks on R.
   We have $\sum_{v\in R} E[\widetilde{X}(v)] = O(B/\epsilon) = O(1/\epsilon) = O(\epsilon^{-1})$ by Lemma 4. By definition of Block Policies, any single-point block v in $T_{\widetilde{\sigma}}$ satisfies $E[\widetilde{X}(v)] > \epsilon^{13}$. Then there are at most $O(\epsilon^{-1})/\epsilon^{13} = O(\epsilon^{-14})$ single-point in path R.
   By definition of Discretization, any $\widetilde{X}(v)$ with non-zero size after discretization, is no less than ϵ^4. Then there are at most $O(1/\epsilon^4) = O(\epsilon^{-4})$ nodes w corresponding to a non-zero size realization.
   This gives a bound $O(\epsilon^{-14} + \epsilon^{-4}) = O(\epsilon^{-14})$ blocks on any root-leaf path in the decision tree.
2. After Discretization in Sect. 4.2, we have only $O(B/\epsilon^5)$ number of values which means there are $|S| = O(B/\epsilon^5)$ children for each block.
3. This is exactly what we define our blocks.

And then we prove Theorem 3.

Proof (The proof of Theorem 3). By Properties 1,2 in Lemma 6 above, there are only constant number of blocks in the decision tree $T_{\widetilde{\sigma}}$ with block policies. Since there are at most $O(\epsilon^{-14})$ blocks in the optimal policy path in decision tree $T_{\widetilde{\sigma}}$ of reward $(1 - O(\epsilon))\mathrm{OPT}$, there exists a block M with $\Omega(\epsilon^{14}\mathrm{OPT})$ reward.

We consider two situations in block M with different number of nodes.

Block M is a single-node block of node v.

And the node v has a huge reward. Then the single-node tour from the starting point ρ to v has an expected reward $\Omega(\epsilon^{14}\mathrm{OPT})$.

Otherwise, block M has at least two nodes.

And block M has an expected reward $\Omega(\epsilon^{14}\mathrm{OPT})$. By Property 3 in Lemma 6, we have $\sum_{b\in M} E[\widetilde{X_b}] \le \epsilon^{13}$. Since the distance from the starting point $\widetilde{\rho}$ to the first node in M is at most 1, the path $P^\star$ which connects $\widetilde{\rho}$ and the block M in order has length at most $1 + \epsilon^{13}$. Thus $P^\star$ is a feasible path for $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ as defined in Definition 2, and it has an expected reward $\Omega(\epsilon^{14}\text{OPT}) \leq \text{OPT}_{\text{TO}}$.

Thus, either there is a single-point tour with expected reward $\Omega(\text{OPT})$, or the valid Task Orienteering instance $I_{\text{TO}}(\epsilon, I_{\text{SO}})$ has reward $\text{OPT}_{\text{TO}} = \Omega(\text{OPT})$, with constant approximation parameter ϵ^{14}. And this completes the proof for Theorem 3.

5 Conclusion

We describe an $O(1)$-approximation algorithm with time budget $(1+\epsilon)B$. As the expected reward of the optimal policy at time budget B is at most $O(\epsilon^{-14})$ times the expected reward of our [time $(1+\epsilon)B$] policy, we get an $O(1)$-approximation for any constant $0 < \epsilon < 1$.

References

1. Arkin, E.M., Mitchell, J.S.B., Narasimhan, G.: Resource-constrained geometric network optimization. In: Proceedings of the Fourteenth Annual Symposium on Computational Geometry, pp. 307–316 (1998)
2. Avrim, B., Shuchi, C., Karger David, R., Terran, L., Adam, M., Maria, M.: Approximation algorithms for orienteering and discounted-reward TSP. SIAM J. Comput. **37**(2), 653 (2007)
3. Bansal, N., Nagarajan, V.: On the Adaptivity Gap of Stochastic Orienteering. Springer International Publishing, Cham (2014)
4. Bhalgat, A., Goel, A., Khanna, S.: Improved approximation results for stochastic knapsack problems. In: Proceedings of the Twenty Second Annual ACM SIAM Symposium on Discrete Algorithms, vol. 27(2), pp. 1647–1665 (2011)
5. Bienstock, D., Goemans, M.X., Simchi-Levi, D., Williamson, D.: A note on the prize collecting traveling salesman problem. Math. Program. **59**(3), 413–420 (1993)
6. Chekuri, C., Korula, N., Pal, M.: Improved algorithms for orienteering and related problems. ACM Trans. Algorithms **8**(3), 661–670 (2008)
7. Chen, K., Har-Peled, S.: The orienteering problem in the plane revisited. In: Proceedings of Symposium on Computational Geometry, pp. 247–254 (2006)
8. Golden, B.L., Levy, L., Vohra, R.: The orienteering problem. Nav. Res. Logist. **34**(34), 307–318 (1987)
9. Gupta, A., Krishnaswamy, R., Nagarajan, V., Ravi, R.: Approximation algorithms for stochastic orienteering. In: Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1522–1538 (2012)
10. Li, J., Yuan, W.: Stochastic combinatorial optimization via poisson approximation. In: Proceedings of the Annual ACM Symposium on Theory of Computing, pp. 971–980 (2012)
11. Ross, K.W., Tsang, D.H.K.: Stochastic knapsack problem. IEEE Trans. Commun. **37**(7), 740–747 (1989)
12. Tsiligirides, T.: Heuristic methods applied to orienteering. J. Oper. Res. Soc. **35**(9), 797–809 (1984)
13. Vansteenwegen, P., Souffriau, W., Van Oudheusden, D.: The orienteering problem: a survey. Mitteilungen Klosterneuburg Rebe Und Wein Obstbau Und Fruchteverwertung **209**(209), 1–10 (2011)