Approximation Algorithms for Optimal Decision Trees and Adaptive TSP Problems

Anupam Gupta[1,⋆], Viswanath Nagarajan[2], and R. Ravi[3,⋆⋆]

[1] Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[2] IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA
[3] Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Abstract. We consider the problem of constructing optimal decision trees: given a collection of tests which can disambiguate between a set of m possible diseases, each test having a cost, and the a-priori likelihood of the patient having any particular disease, what is a good adaptive strategy to perform these tests to minimize the expected cost to identify the disease? We settle the approximability of this problem by giving a tight $O(\log m)$-approximation algorithm.

We also consider a more substantial generalization, the *Adaptive TSP* problem, which can be used to model switching costs between tests in the optimal decision tree problem. Given an underlying metric space, a random subset S of cities is drawn from a known distribution, but S is initially unknown to us—we get information about whether any city is in S only when we visit the city in question. What is a good adaptive way of visiting all the cities in the random subset S while minimizing the expected distance traveled? For this adaptive TSP problem, we give the first poly-logarithmic approximation, and show that this algorithm is best possible unless we can improve the approximation guarantees for the well-known group Steiner tree problem.

1 Introduction

Consider the following two adaptive covering optimization problems:

- *Adaptive TSP under stochastic demands.* (AdapTSP) A traveling salesperson is given a metric space (V, d), distinct subsets $S_1, S_2, \ldots, S_m \subseteq V$ such that S_i appears with probability p_i (and $\sum_i p_i = 1$), and she needs to serve requests at this subset of locations. However, she does not know the identity of the random subset: she can only visit locations, at which time she finds out whether or not that location is part of the subset. What adaptive strategy should she use to minimize the expected time to serve all requests?
- *Optimal Decision Trees.* Given a set of m diseases, there are n binary tests that can be used to disambiguate between these diseases. If the cost of

⋆ Supported in part by NSF awards CCF-0448095 and CCF-0729022, and an Alfred P. Sloan Fellowship.
⋆⋆ Supported in part by NSF grant CCF-0728841.

S. Abramsky et al. (Eds.): ICALP 2010, Part I, LNCS 6198, pp. 690–701, 2010.
© Springer-Verlag Berlin Heidelberg 2010

performing test $t \in [n]$ is c_t, and we are given the likelihoods $\{p_j\}_{j\in[m]}$ that a typical patient has the disease j, what (adaptive) strategy should the doctor use for the tests to minimize the expected cost to identify the disease?

It can be shown that the optimal decision tree problem is a special case of the former problem (a formal reduction is given in Section 4.) In both these problems we want to devise adaptive strategies, which take into account the revealed information in the queries so far (e.g., cities already visited, or tests already done) to determine the future course of action—i.e., our output must be a decision tree. In contrast, a non-adaptive strategy corresponds to a decision list. While some problems have the property that the best adaptive and non-adaptive strategies have similar performances, both the above problems have a large *adaptivity gap* [9] (the worst ratio between the performance of the optimal non-adaptive and optimal adaptive strategies). Hence it is essential that we seek good *adaptive* strategies.

The *optimal decision tree problem* has long been studied (its NP-hardness was shown in 1976 [22], and many references and applications can be found in [29]). On the algorithms side, $O(\log m)$-approximation algorithms have been given for the special cases where the likelihoods $\{p_j\}$ are all equal, or where the test costs $\{c_t\}$ are all equal [25,8,1,4,29,18]. But the problem has remained open for the case when both the probabilities and the costs are arbitrary—the previous results only imply $O\left(\log \frac{1}{p_{\min}}\right)$ and $O\left(\log(m\,\frac{c_{max}}{c_{min}})\right)$ approximation algorithms for the general case. An $O(\log m)$-approximation for general costs and probabilities would be the best possible unless $NP \subseteq DTIME[n^{O(\log\log n)}]$ [4]; and while the existence of such an algorithm has been posed as an open question, it has not been answered prior to this work.

Apart from being a natural adaptive optimization problem, AdapTSP can be viewed as a generalization of the optimal decision tree problem when there are arbitrary (metric) switching costs between tests. Similar extensions from uniform to metric switching-costs, have been studied for the *multi-armed bandit* problem [15,16]. To the best of our knowledge, the adaptive TSP problem has not been studied previously.

In this paper, we settle the approximability of the optimal decision tree problem:

Theorem 1. *There is an $O(\log m)$-approximation for the optimal decision tree problem with arbitrary test costs and arbitrary probabilities; the problem admits the same approximation ratio even when the tests have non-binary outcomes.*

In fact, this result arises as a special case of the following theorem:

Theorem 2. *There is an $O(\log^2 n \log m)$-factor approximation algorithm for the adaptive TSP problem.*

We also show that AdapTSP is at least as hard to approximate as the well-known group Steiner tree problem [12,21]. Thus AdapTSP is $\Omega(\log^{2-\epsilon} n)$ hard to approximate even on tree metrics; our results are essentially best possible on such metrics, and we lose an extra logarithmic factor to go to general metrics, as in the group Steiner tree problem.

A Word about our Techniques. To solve the AdapTSP problem, we first solve the "Isolation" problem, which seeks to identify which of the m scenarios has materialized; once we know the scenario we can visit its vertices using, say, Christofides' heuristic. The idea behind our algorithm for Isolation is this—suppose each vertex lies in at most half the scenarios; if we visit one vertex in each of the m scenarios using a short tour (which is just group Steiner tree [12]), we'd notice at least one of these vertices to have a demand—this would reduce the number of possible scenarios by 50%. This is necessarily an over-simplified view, and there are many details to handle: we need not visit all scenarios—visiting all but one allows us to infer the last one by exclusion; the expectation in the objective function means we need to solve a *min-latency* version of group Steiner tree; not all vertices need lie in less than half the scenarios. Another major issue is that moreover we do not want our performance to depend on the size of the probabilities, in case some of them are exponentially small, so we cannot just hope to reduce the measure of the remaining scenarios by 50%. Finally, we need to charge our cost directly against the optimal decision tree. All these issues can be resolved to give an $O(\log^2 n \cdot \log m)$ approximation for Isolation (and hence for AdapTSP) with n nodes and m scenarios. The Isolation algorithm also involves an interesting combination of ideas from the group Steiner [12,5] and minimum latency [2,6,10] problems—it uses a greedy strategy that is greedy with respect to two different criteria, namely both the probability measure and the number of scenarios. This idea is formalized in our definition of the *partial latency* group Steiner (LPGST) problem.

For the special case of the optimal decision tree problem, we show that we can use the min-sum set cover problem [11] instead of the min-latency version of group Steiner tree; this avoids an $O(\log^2 n)$ loss in the approximation guarantee, and hence gives us an optimal $O(\log m)$-approximation for the optimal decision tree problem. Our result further reinforces the close connection between the min-sum set cover problem and the optimal decision tree problem that was first noticed by Chakravarthy et al. [4].

Paper Outline. The results on AdapTSP and Isolation appear in Section 3, and the improved algorithm for optimal decision tree appears in Section 4. Due to lack of space, we refer the reader to the full version [20] for all missing proofs. In [20] we also give the hardness result for AdapTSP, and extend our algorithm to obtain a poly-logarithmic approximation algorithm for the related *adaptive traveling repairman problem* (where the objective is expected response time).

1.1 Other Related Work

The optimal decision tree problem has been studied earlier by many authors, with algorithms and hardness results being shown by [22,25,1,8,4,3,18]; as mentioned above, the algorithms in these papers give $O(\log m)$-approximations only when either the probabilities or costs (or both) are polynomially-bounded. Table 1 in Guillory and Bilmes [18] gives a good summary of known approximation ratios; in general, the best approximation guarantees are $O\left(\log \frac{1}{p_{\min}}\right)$ and

$O\left(\log(m\,\frac{c_{max}}{c_{min}})\right)$. The $O(\log m)$-approximation for arbitrary costs and probabilities solves an open problem from these papers.

There are many results on adaptive optimization. E.g., [13] considered an adaptive set-cover problem; they give an $O(\log n)$-approximation when sets may be chosen multiple times, and an $O(n)$-approximation when each set may be chosen at most once. This was improved in [27] to $O(\log^2 n \log m)$, and subsequently to the best-possible $O(\log n)$-approximation by [26], also using a greedy algorithm. In very recent work [14] generalize adaptive set-cover to so-called 'adaptive submodularity', and give some applications. [7] considered adaptive strategies for stochastic routing problems using a greedy-like algorithm where they solved an LP at each step. [17] studied adaptive transmission in noisy channels (that have multiple states) to maximize the utility (i.e., the difference between success-probability and probing-cost). The adaptivity-gap is large in all these problems, as is the case for the problems considered in this paper, and hence the solutions need to be inherently adaptive.

The AdapTSP problem is related to universal TSP [24,19] and *a priori* TSP [23,30,31] only in spirit—in both the universal and *a priori* TSP problems, we seek a master tour which we shortcut once the demand set is known, and the goal is to minimize the worst-case or expected length of the shortcut tour. The crucial difference is that the demand subset is revealed *in toto* in these two problems, leaving no possibility of adaptivity—this is in contrast to the slow revelation of the demand subset that occurs in AdapTSP.

2 Notation

Throughout this paper, we deal with demand distributions over vertex-subsets that are specified explicitly. I.e. demand distribution $\mathcal{D}$ is specified by m distinct subsets $\{S_i \subseteq V\}_{i=1}^m$ having associated probabilities $\{p_i\}_{i=1}^m$ such that $\sum_{i=1}^m p_i = 1$. This means that the realized subset $D \subseteq V$ of demand-vertices will always be one of $\{S_i\}_{i=1}^m$, where $D = S_i$ with probability p_i (for all $i \in [m]$). We also refer to the subsets $\{S_i\}_{i=1}^m$ as *scenarios*. The following definition captures all adaptive strategies.

Definition 1 (Decision Tree). *A decision tree T in metric (V, d) is a rooted binary tree where each non-leaf node of T is labeled with a vertex $u \in V$, and its two children u_{yes} and u_{no} correspond to the subtrees taken if there **is** demand at u or if there is **no** demand at u. Thus given any realized demand $D \subseteq V$, a unique path P_D is followed in T from the root down to a leaf.*

For metric (V, d) and root $r \in V$, an *r-tour* is a tour that begins and ends at r.

3 The Adaptive TSP and Isolation Problems

The input to *adaptive TSP* is a metric (V, d) with root $r \in V$ and a demand distribution $\mathcal{D}$ over subsets of vertices. The crucial aspect in this model is that information on whether or not there is demand at a vertex v is obtained only

when that vertex v is visited. The objective is to find an adaptive strategy that minimizes the expected time to visit all vertices of the realized scenario drawn from $\mathcal{D}$.

We assume that the distribution $\mathcal{D}$ is specified *explicitly* with a support-size of m. The more general setting would be to consider black-box access to distribution $\mathcal{D}$: however, as shown in [28], AdapTSP under general black-box distributions admits no $o(n)$ approximation if the algorithm is restricted to polynomially many samples. This is the reason we consider an explicit demands model for $\mathcal{D}$. Moreover, AdapTSP under explicit demands still contains interesting special cases such as optimal decision tree problem. One could also consider AdapTSP under independent demand distributions; however in this case there is a trivial solution: visit all vertices having non-zero probability along an approximately minimum TSP tour. In terms of Definition 1, we have:

Definition 2 (The AdapTSP Problem). *Given metric (V, d), root r and demand distribution $\mathcal{D}$, the goal in* AdapTSP *is to compute a decision tree T in metric (V, d) with the added conditions that:*

- *the root of T is labeled with the root vertex r, and*
- *for each scenario $i \in [m]$, the path P_{S_i} followed on input $S_i \in support(\mathcal{D})$ contains <u>all</u> vertices in S_i.*

The objective function is to minimize the expected tour length $\sum_{i=1}^{m} p_i \cdot d(P_{S_i})$, where $d(P_{S_i})$ is the length of the tour that starts at r, visits the vertices on path P_{S_i} in that order, and returns to r.

A closely related problem is the *Isolation* problem. This has the same input as the AdapTSP problem, but the goal is not necessarily to visit all the vertices in the realized scenario, but just to identify the unique scenario that has materialized.

Definition 3 (The Isolation Problem). *Given metric (V, d), root r and demand distribution $\mathcal{D}$, the goal in* Isolation *is to compute a decision tree T in metric (V, d) with the added conditions that:*

- *the root of T is labeled with the root vertex r, and*
- *for each scenario $i \in [m]$, the path P_{S_i} followed on input $S_i \in support(\mathcal{D})$ ends at a <u>distinct</u> leaf-node of T.*

The objective function is to minimize the expected tour length $\mathsf{IsoTime}(T) := \sum_{i=1}^{m} p_i \cdot d(P_{S_i})$, where $d(P_{S_i})$ is the length of the tour that starts at r, visits the vertices on path P_{S_i} in that order, and returns to r.

Note the only difference between this definition and the one for AdapTSP is that the tree path P_{S_i} need not contain all vertices of S_i, and the paths for different scenarios should end at distinct leaf-nodes. The following simple lemma relates these two objectives.

Lemma 1. *An α-approximation algorithm for* Isolation *implies an $\left(\alpha + \frac{3}{2}\right)$ approximation for* AdapTSP.

In the next few subsections, we give an $O(\log^2 n \log m)$-approximation algorithm for Isolation, which by this lemma implies the same result for AdapTSP.

3.1 Approximation Algorithm for the Isolation Problem

Recall that an instance of Isolation is specified by a metric (V, d), a root vertex $r \in V$, and m scenarios $\{S_i\}_{i=1}^m$ with associated probability values $\{p_i\}_{i=1}^m$. At a high level, our approach is a simple divide-and-conquer based one. The algorithm for Isolation is recursive: given an instance, it first develops a strategy to generate several sub-instances from this instance, each given by some proper subset $M \subseteq [m]$ of scenarios, with associated probabilities $\{q_i\}_{i \in M}$ where $\sum_{i \in M} q_i = 1$. (We refer to such a sub-instance of the original Isolation instance as $\langle M, \{q_i\}_{i \in M} \rangle$.) We then recursively build an isolation strategy within each sub-instance. The real question is: *How do we generate these sub-instances so that we can charge these to the cost of the best decision tree?*

A useful subroutine to address this and solve Isolation will be the *partial latency group Steiner* (LPGST) problem, where we are given a metric (V, d), g groups of vertices $\{X_i \subseteq V\}_{i=1}^g$ with associated weights $\{w_i\}_{i=1}^g$, root $r \in V$, and a target $h \leq g$. A group $i \in [g]$ is said to be *covered* (or visited) by r-tour τ if any vertex in X_i is visited, and the *arrival time* of group i is the length of the shortest prefix of τ that contains an X_i-vertex. The arrival times of all uncovered groups are set to be the tour-length. The weighted sum of arrival times of all groups is termed *latency* of the tour, i.e.,

$$\text{latency}(\tau) = \textstyle\sum_{i \text{ covered}} w_i \cdot \text{arrival time}_\tau(X_i) + \sum_{i \text{ uncovered}} w_i \cdot \text{length}(\tau). \quad (3.1)$$

The goal in the LPGST problem is to compute a *minimum latency* tour that *covers at least* h groups. Note that this objective generalizes the standard min-latency objective, where we have to cover *all* groups. In the full version [20] we prove the following result:

Theorem 3. *There is an* $(O(\log^2 n), 4)$*-bicriteria approximation algorithm for* LPGST*. I.e., the tour output by the algorithm visits at least* $\frac{h}{4}$ *groups and has latency at most* $O(\log^2 n)$ *times the optimal latency of a tour that visits* h *groups.*

In this subsection, we present the algorithm for Isolation assuming this result for LPGST. Let us begin by showing the partitioning algorithm.

The Partitioning Algorithm. The algorithm Partition (given below as Algorithm 1) finds an r-tour τ in the metric such that after observing the demands on τ, the *number of scenarios* that are consistent with these observations is only a constant fraction of the total. E.g., if there was a vertex v such that $\approx 50\%$ of the scenarios contained it, then visiting vertex v would reduce the candidate scenarios by $\approx 50\%$, irrespective of the observation at v, giving us a tangible notion of progress. However, each vertex may give a very unbalanced partition or such a vertex may be too expensive to visit, so we may have to visit multiple vertices—this is the basic idea in algorithm Partition.

The Isolation Algorithm. To follow a divide-and-conquer strategy, the final algorithm is fairly natural given the partitioning scheme. The algorithm IsoAlg (given as Algorithm 2) proceeds in several phases. In each phase, it maintains

Algorithm 1. Algorithm Partition($\langle M, \{q_i\}_{i \in M} \rangle$)

1: **let** $g = |M|$. For each $v \in V$, define $F_v := \{i \in M \mid v \in S_i\}$, and
$D_v := \begin{cases} F_v & \text{if } |F_v| \leq \frac{g}{2} \\ M \setminus F_v & \text{if } |F_v| > \frac{g}{2} \end{cases}$
2: **for each** $i \in M$, set $X_i \leftarrow \{v \in V \mid i \in D_v\}$. Note that either the presence of a demand at some vertex v reduces the number of still-possible scenarios by half, or the absence of demand does so. To handle this asymmetry, this step takes scenarios $\{S_i\}_{i \in M}$ and "flips" some vertices to get X_i.
3: **run** the LPGST algorithm (Theorem 3) with metric (V, d) with root r, groups $\{X_i\}_{i \in M}$ with weights $\{q_i\}_{i \in M}$, and target $|M| - 1$.
let $\tau := r, v_1, v_2, \cdots, v_{t-1}, r$ be the r-tour returned.
4: **let** $\{P_k\}_{k=1}^{t}$ be the partition of M where
$P_k := \begin{cases} D_{v_k} \setminus \left(\cup_{j<k} D_{v_j}\right) & \text{if } 1 \leq k \leq t-1 \\ M \setminus \left(\cup_{j<t} D_{v_j}\right) & \text{if } k = t \end{cases}$
5: **return** tour τ and the partition $\{P_k\}_{k=1}^{t}$.

a candidate set M of scenarios such that the realized scenario lies in M. Upon observing demands along the tour produced by algorithm Partition (in Step 2), a new set $M' \subseteq M$ containing the realized scenario is identified such that the number of candidate scenarios reduces by a constant factor (i.e. $|M'| \leq \frac{7}{8} \cdot |M|$); then IsoAlg recurses on scenarios M'. After $O(\log m)$ such phases the realized scenario would be correctly identified.

Algorithm 2. Algorithm IsoAlg$\langle M, \{q_i\}_{i \in M} \rangle$

1: If $|M| = 1$, return this unique scenario as realized.
2: **run** Partition$\langle M, \{q_i\}_{i \in M} \rangle$
let $\tau = (r, v_1, v_2, \cdots, v_{t-1}, r)$ be the r-tour and $\{P_k\}_{k=1}^{t}$ be the partition of M returned.
3: **let** $q'_j := \sum_{i \in P_k} q_i$ **for all** $j \in 1 \ldots t$.
4: **traverse** tour τ and return directly to r after visiting the first (if any) vertex v_{k^*} (for $k^* \in [t-1]$) that determines that the realized scenario is in $P_{k^*} \subseteq M$. If there is no such vertex until the end of the tour τ, then set $k^* \leftarrow t$.
5: **run** IsoAlg$\langle P_{k^*}, \{\frac{q_i}{q'_{k^*}}\}_{i \in P_{k^*}} \rangle$ to isolate the realized scenario within the subset P_{k^*}.

Note that the adaptive Algorithm IsoAlg implicitly defines a decision tree too: create a path $(r, v_1, v_2, \cdots, v_{t-1}, v_t = r)$, and hang the subtrees created in the recursive call on each instance $\langle P_k, \{\frac{q_i}{q'_k}\} \rangle$ from the respective node v_k.

Remark: We use the LPGST problem as a subroutine in solving Isolation, instead of the seemingly simpler *density group Steiner* problem [5]. The reason behind this is that we measure progress in the recursive scheme IsoAlg as the *number* of candidate scenarios, whereas in computing the objective we need to use the *probabilities* of scenarios. This is also the reason why our final algorithm interleaves both greedy objectives: number and probabilities of scenarios. It is

not clear whether a greedy algorithm w.r.t. only one of these objectives can achieve the same approximation ratio.

3.2 The Analysis for Algorithm IsoAlg

We now prove the performance guarantee and correctness of Algorithm IsoAlg. By the construction in algorithm IsoAlg, it follows that the realized scenario is identified after $O(\log m)$ recursive calls to IsoAlg. To complete the analysis, we need two additional ideas: (1) relating the LPGST and Isolation objective-values in each call to IsoAlg (which bounds the cost in a single sub-instance), and (2) establishing a subadditivity property of Isolation, that bounds the expected cost across all 'parallel' sub-instances (i.e. those in the same phase).

For any instance $\mathcal{J}$ of the isolation problem, let $\mathsf{IsoTime}^*(\mathcal{J})$ denote its optimal value. Missing proofs in the following analysis can be found in [20].

Claim 1. *For any instance $\mathcal{J} = \langle M, \{q_i\}_{i \in M} \rangle$, the optimal value of the* LPGST *instance considered in Step 3 of Algorithm* Partition($\mathcal{J}$) *is at most* $\mathsf{IsoTime}^*(\mathcal{J})$*— i.e., the* LPGST *instance costs at most the isolation instance.*

Proof: Let T be an optimal decision tree corresponding to Isolation instance $\mathcal{J}$, and hence $\mathsf{IsoTime}^*(\mathcal{J}) = \mathsf{IsoTime}(T)$. Note that by definition of the sets $\{F_v, D_v\}_{v \in V}$, any internal node in T labeled vertex v has its two children v_{yes} and v_{no} corresponding to the realized scenario being in either F_v or $M \setminus F_v$ (in that order), or equivalently, D_v or $M \setminus D_v$ (now not necessarily in that order).

Consider an r-tour σ that starts at the root r of T and moves from each node v to its unique child that corresponds to $M \setminus D_v$, until it reaches a leaf-node l, when it returns to r. Let the vertices in this tour σ be $r, u_1, u_2, \cdots, u_j, r$ (where u_j is the last vertex visited in T before leaf l). Since T is a feasible decision tree for the isolation instance, the leaf l is labeled by a unique scenario $a \in M$ such that when T is run under demands S_a, it traces path from the root to leaf node l. Moreover, every scenario $b \in M \setminus \{a\}$ gives rise to a root-leaf path that diverges from the root-l path (i.e. σ). From the way we constructed σ, the scenarios that diverged from σ were precisely $\cup_{k=1}^{j} D_{u_k}$, and hence $\cup_{k=1}^{j} D_{u_k} = M \setminus \{a\}$.

Now consider the r-tour σ as a potential solution to the LPGST instance in Step 3. Since $|\cup_{k=1}^{j} D_{u_k}| = |M| - 1$, the definition of the X_i's says that this path hits $|M| - 1$ of these sets X_i, so it is indeed feasible. Also, the definition of the isolation cost implies that the latency of this tour σ is at most the isolation cost $\mathsf{IsoTime}(T) = \mathsf{IsoTime}^*(\mathcal{J})$. ■

If we use a $(\rho_{\mathsf{LPGST}}, 4)$-bicriteria LPGST algorithm, we get the following claim:

Claim 2. *The latency of tour τ returned by Algorithm* Partition *is at most $\rho_{\mathsf{LPGST}} \cdot \mathsf{IsoTime}^*(\langle M, \{q_i\}_{i \in M} \rangle)$. Furthermore, the resulting partition $\{P_k\}_{k=1}^{t}$ has each $|P_k| \leq \frac{7}{8}|M|$ for each $k \in [t]$, when $|M| \geq 2$.*

Proof: By Claim 1, the optimal value of the LPGST instance in Step 3 of algorithm Partition is at most $\mathsf{IsoTime}^*(\langle M, \{q_i\}_{i \in M} \rangle)$; now the approximation guarantee from Theorem 3 implies that the latency of the solution tour τ is at most ρ_{LPGST} times that. This proves the first part of the claim.

Consider $\tau := \langle r = v_0, v_1, \cdots, v_{t-1}, v_t = r \rangle$ the tour returned by the LPGST algorithm in Step (3) of algorithm Partition; and $\{P_k\}_{k=1}^t$ the resulting partition. Claim 1 and Theorem 3 imply that the number of groups covered by τ is $|\cup_{k=1}^{t-1} D_{v_k}| \geq \frac{|M|-1}{4} \geq \frac{|M|}{8}$ (when $|M| \geq 2$). By definition of the sets D_v, it holds that $|D_v| \leq |M|/2$ for all $v \in V$; moreover, since all but the last part P_k is a subset of some D_v, it holds that $|P_k| \leq \frac{|M|}{2}$ for $1 \leq k \leq t-1$. Moreover, the set P_t has size $|P_t| = |M \setminus (\cup_{j<t} D_{v_j})| \leq \frac{7}{8}|M|$. ■

Of course, we don't really care about the latency of the tour *per se*, we care about the cost incurred in isolating the realized scenario. But the two are related (by their very construction), as the following claim formalizes:

Claim 3. *At the end of Step 4 of* $\mathsf{IsoAlg}\langle M, \{q_i\}_{i \in M}\rangle$*, the realized scenario lies in* P_{k^*}*. The expected distance traversed in this step* $\leq 2\rho_{\mathsf{LPGST}} \cdot \mathsf{IsoTime}^*(\langle M, \{q_i\}_{i \in M}\rangle)$.

Now, the following simple claim captures the "sub-additivity" of $\mathsf{IsoTime}^*$.

Claim 4. *For any instance* $\langle M, \{q_i\}_{i \in M}\rangle$ *and any partition* $\{P_k\}_{k=1}^t$ *of* M*,*

$$\sum_{k=1}^t q'_k \cdot \mathsf{IsoTime}^*(\langle P_k, \{\tfrac{q_i}{q'_k}\}_{i \in P_k}\rangle) \leq \mathsf{IsoTime}^*(\langle M, \{q_i\}_{i \in M}\rangle), \tag{3.2}$$

where $q'_k = \sum_{i \in P_k} q_i$ *for all* $1 \leq k \leq t$.

Proof: Let T denote the optimal decision tree for the instance $\mathcal{J}_0 := \langle M, \{q_i\}_{i \in M}\rangle$. For each $k \in [t]$, consider instance $\mathcal{J}_k := \langle P_k, \{\frac{q_i}{q'_k}\}_{i \in P_k}\rangle$; one feasible adaptive strategy for instance $\mathcal{J}_k$ is obtained by taking the decision tree T and considering only paths to the leaf-nodes labeled by $\{i \in P_k\}$. Note that this is a feasible solution since T isolates all scenarios $\cup_{k=1}^t P_k$. Moreover, the expected cost of such a strategy for $\mathcal{J}_k$ is $\sum_{i \in P_k} \frac{q_i}{q'_k} \cdot d(\pi_i)$ where π_i denotes the tour traced by T under scenario $i \in P_k$. Hence $\mathsf{Opt}(\mathcal{J}_k) \leq \sum_{i \in P_k} \frac{q_i}{q'_k} \cdot d(\pi_i)$. Summing over all parts $k \in [t]$, we get

$$\sum_{k=1}^t q'_k \cdot \mathsf{Opt}(\mathcal{J}_k) \leq \sum_{k=1}^t q'_k \cdot \sum_{i \in P_k} \frac{q_i}{q'_k} \cdot d(\pi_i) = \sum_{i \in M} q_i \cdot d(\pi_i) = \mathsf{Opt}(\mathcal{J}_0), \tag{3.3}$$

where the penultimate equality uses the fact that $\{P_k\}_{k=1}^t$ partitions M. ■

Given the above claims, we finally bound the expected cost of the strategy given by our algorithm.

Theorem 4. *The expected length of the strategy given by* $\mathsf{IsoAlg}\langle M, \{q_i\}_{i \in M}\rangle$ *is at most:*

$$2\rho_{\mathsf{LPGST}} \cdot \log_{8/7} |M| \cdot \mathsf{IsoTime}^*(\langle M, \{q_i\}_{i \in M}\rangle).$$

Proof: We prove this by induction on $|M|$. The base case of $|M| = 1$ is trivial, since zero length is traversed, and hence we consider $|M| \geq 2$. Let instance $\mathcal{I}_0 := \langle M, \{q_i\}_{i \in M}\rangle$. For $k \in [t]$, consider the sub-instance $\mathcal{I}_k := \langle P_k, \{\frac{q_i}{q'_k}\}_{i \in P_k}\rangle$, where $q'_k = \sum_{i \in P_k} q_i$. By the inductive hypothesis, for any $k \in [t]$, the expected

length of $\mathsf{IsoAlg}(\mathcal{I}_k)$ is at most $2\rho_{\mathsf{LPGST}} \cdot \log_{8/7}|P_k| \cdot \mathsf{IsoTime}^*(\mathcal{I}_k) \leq 2\rho_{\mathsf{LPGST}} \cdot (\log_{8/7}|M| - 1) \cdot \mathsf{IsoTime}^*(\mathcal{I}_k)$, since $|P_k| \leq \frac{7}{8}|M|$ (from Claim 2 as $|M| \geq 2$).

By Claim 3, the expected length traversed in Step 4 of $\mathsf{IsoAlg}(\mathcal{I}_0)$ is at most $2\rho_{\mathsf{LPGST}} \cdot \mathsf{IsoTime}^*(\mathcal{I}_0)$. The probability of recursing on $\mathcal{I}_k$ is exactly q'_k for each $k \in [t]$, hence the expected length of $\mathsf{IsoAlg}(\mathcal{I}_0)$ is at most:

$$\begin{aligned}
&2\rho_{\mathsf{LPGST}} \cdot \mathsf{IsoTime}^*(\mathcal{I}_0) + \textstyle\sum_{k=1}^{t} q'_k \cdot (\text{exp. length of } \mathsf{IsoAlg}(\mathcal{I}_k)) \\
&\leq 2\rho_{\mathsf{LPGST}} \cdot \mathsf{IsoTime}^*(\mathcal{I}_0) + \textstyle\sum_{k=1}^{t} q'_k \cdot 2\rho_{\mathsf{LPGST}} \cdot (\log_{8/7}|M| - 1) \cdot \mathsf{IsoTime}^*(\mathcal{I}_k) \\
&\leq 2\rho_{\mathsf{LPGST}} \cdot \mathsf{IsoTime}^*(\mathcal{I}_0) + 2\rho_{\mathsf{LPGST}} \cdot (\log_{8/7}|M| - 1) \cdot \mathsf{IsoTime}^*(\mathcal{I}_0) \\
&= 2\rho_{\mathsf{LPGST}} \cdot \log_{8/7}|M| \cdot \mathsf{IsoTime}^*(\mathcal{I}_0)
\end{aligned}$$

where the third inequality uses Claim 4. ■

Applying Theorem 4 on the original Isolation instance gives an $O(\rho_{\mathsf{LPGST}} \cdot \log m)$-approximately optimal algorithm; using Theorem 3 implies $\rho_{\mathsf{LPGST}} = O(\log^2 n)$.

Theorem 5. *There is an $O(\log^2 n \cdot \log m)$-approximation algorithm for* Isolation *with n vertices and m scenarios.*

Combining this theorem with Lemma 1 we obtain Theorem 2. A comment about the optimality of these results: we show in [20] that AdapTSP (and by Lemma 1, Isolation as well) is as hard to approximate as the group Steiner problem, and hence any improvement here will make progress on that problem too.

4 Optimal Decision Tree Problem

In the *optimal decision tree problem* [25,1,3], we are given a set of m diseases with associated probabilities $\{p_i\}_{i=1}^m$ that sum to 1, and a collection $\{T_j\}_{j=1}^n$ of n binary tests with non-negative costs $\{c_j\}_{j=1}^n$. Each test $j \in [n]$ is a subset T_j of the diseases that correspond to passing the test; so performing test j distinguishes between diseases T_j and $[m] \setminus T_j$.

Definition 4. *A* test strategy *S is a binary tree where each internal node is labeled by a test, and each leaf node is labeled by a disease such that:*

- *For each disease $i \in [m]$ define a path π_i in S from the root node to some leaf as follows. At any internal node, if i passes the test then π_i follows the right branch; if it fails the test then π_i follows the left branch.*
- *For each $i \in [m]$, the path π_i ends at a leaf labeled disease i.*

The cost *L_i of a disease $i \in [m]$ is the sum of test-costs along path π_i; and the cost of the test strategy S is $\sum_{i=1}^m p_i \cdot L_i$. The objective in the optimal decision tree problem is to compute a test strategy of minimum cost.*

Observe that the optimal decision tree problem is a special case of Isolation: given an instance of optimal decision tree (as above), consider a metric (V, d) induced by a weighted star with center r and n leaves corresponding to the tests. For each $j \in [n]$, we set $d(r, j) = \frac{c_j}{2}$. The demand scenarios are as follows: for each $i \in [m]$, scenario i is $\{j \in [n] \mid i \in T_j\}$. It is easy to see that this Isolation instance corresponds exactly to the optimal decision tree instance. Based on

this, it suffices to focus on Isolation in weighted star-metrics, and we obtain the following improved bound (see [20]).

Theorem 6. *There is an* $\big(O(1), O(1)\big)$ *bicriteria approximation algorithm for* LPGST *on weighted star metrics.*

Setting $\rho_{\mathsf{LPGST}} = O(1)$ (from Theorem 6) in the analysis of Section 3.2, we obtain an $O(\rho_{\mathsf{LPGST}} \cdot \log m) = O(\log m)$ approximation for Isolation on weighted star-metrics. This completes the proof of Theorem 1 for binary outcomes.

Multiway tests. Chakravarthy et al. [3] considered the optimal decision tree problem when the outcomes of tests are multiway (not just binary), and gave an $O(\log m)$-approximation under unit probabilities and costs. We observe that our algorithm can be easily extended to this problem with non-uniform probabilities and costs. In this setting (when each test has at most l outcomes), any test $j \in [n]$ induces a partition $\{T_j^k\}_{k=1}^{l}$ of $[m]$, and performing test j determines which of the parts the realized disease lies in. Firstly note that this problem is also a special case of Isolation. As before consider a metric (V, d) induced by a weighted star with center r and n leaves corresponding to the tests. For each $j \in [n]$, we set $d(r, j) = \frac{c_j}{2}$. Additionally for each $j \in [n]$, introduce l copies of test-vertex j, labeled $(j, 1), \cdots, (j, l)$, at zero distance from each other. The demand scenarios are defined naturally: for each $i \in [m]$, scenario i is $\{(j, k) \mid i \in T_j^k\}$. Clearly this Isolation instance is equivalent to the (multiway) test strategy instance. Since the resulting metric is still a weighted star (we only made vertex copies), Theorem 6 and the algorithm of Section 3.2 imply an $O(\log m)$-approximation for this multiway decision tree problem. Thus we obtain Theorem 1.

References

1. Adler, M., Heeringa, B.: Approximating Optimal Binary Decision Trees. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 1–9. Springer, Heidelberg (2008)
2. Blum, A., Chalasani, P., Coppersmith, D., Pulleyblank, W.R., Raghavan, P., Sudan, M.: The minimum latency problem. In: STOC, pp. 163–171 (1994)
3. Chakaravarthy, V., Pandit, V., Roy, S., Sabharwal, Y.: Approximating Decision Trees with Multiway Branches. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555, pp. 210–221. Springer, Heidelberg (2009)
4. Chakravarthy, V.T., Pandit, V., Roy, S., Awasthi, P., Mohania, M.: Decision Trees for Entity Identification: Approximation Algorithms and Hardness Results. In: PODS (2007)
5. Charikar, M., Chekuri, C., Goel, A., Guha, S.: Rounding via trees: deterministic approximation algorithms for group Steiner trees and k median. In: STOC, pp. 114–123 (1998)
6. Chaudhuri, K., Godfrey, B., Rao, S., Talwar, K.: Paths, trees, and minimum latency tours. In: FOCS, pp. 36–45 (2003)
7. Chawla, S., Roughgarden, T.: Single-source stochastic routing. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX 2006 and RANDOM 2006. LNCS, vol. 4110, pp. 82–94. Springer, Heidelberg (2006)
8. Dasgupta, S.: Analysis of a greedy active learning strategy. In: Advances in Neural Information Processing Systems, NIPS (2004)

9. Dean, B.C., Goemans, M.X., Vondrák, J.: Approximating the stochastic knapsack problem: The benefit of adaptivity. In: FOCS, pp. 208–217 (2004)
10. Fakcharoenphol, J., Harrelson, C., Rao, S.: The k-traveling repairmen problem. ACM Transactions on Algorithms 3(4) (2007)
11. Feige, U., Lovász, L., Tetali, P.: Approximating min sum set cover. Algorithmica 40(4), 219–234 (2004)
12. Garg, N., Konjevod, G., Ravi, R.: A Polylogarithmic Approximation Algorithm for the Group Steiner Tree Problem. Journal of Algorithms 37(1), 66–84 (2000)
13. Goemans, M., Vondrák, J.: Stochastic covering and adaptivity. In: Correa, J.R., Hevia, A., Kiwi, M. (eds.) LATIN 2006. LNCS, vol. 3887, pp. 532–543. Springer, Heidelberg (2006)
14. Golovin, D., Krause, A.: Adaptive Submodularity: A New Approach to Active Learning and Stochastic Optimization (2010), http://arxiv.org/abs/1003.3967
15. Guha, S., Munagala, K.: Approximation algorithms for budgeted learning problems. In: STOC, pp. 104–113. ACM, New York (2007); Full version as Sequential Design of Experiments via Linear Programming, http://arxiv.org/abs/0805.2630v1
16. Guha, S., Munagala, K.: Multi-armed bandits with metric switching costs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5556, pp. 496–507. Springer, Heidelberg (2009)
17. Guha, S., Munagala, K., Sarkar, S.: Information acquisition and exploitation in multichannel wireless networks. CoRR, abs/0804.1724 (2008)
18. Guillory, A., Bilmes, J.: Average-Case Active Learning with Costs. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 141–155. Springer, Heidelberg (2009)
19. Gupta, A., Hajiaghayi, M.T., Räcke, H.: Oblivious network design. In: SODA, pp. 970–979 (2006)
20. Gupta, A., Nagarajan, V., Ravi, R.: Approximation Algorithms for Optimal Decision Trees and Adaptive TSP Problems (full version). ArXiv (2010), http://arxiv.org/abs/1003.0722
21. Halperin, E., Krauthgamer, R.: Polylogarithmic inapproximability. In: STOC, pp. 585–594 (2003)
22. Hyafil, L., Rivest, R.L.: Constructing optimal binary decision trees is NP-complete. Information Processing Lett. 5(1), 15–17 (1976/1977)
23. Jaillet, P.: A priori solution of a travelling salesman problem in which a random subset of the customers are visited. Operations Research 36(6) (1988)
24. Jia, L., Lin, G., Noubir, G., Rajaraman, R., Sundaram, R.: Universal approximations for TSP, Steiner tree, and set cover. In: STOC, pp. 386–395 (2005)
25. Kosaraju, S.R., Przytycka, T.M., Borgstrom, R.S.: On an Optimal Split Tree Problem. In: Dehne, F., Gupta, A., Sack, J.-R., Tamassia, R. (eds.) WADS 1999. LNCS, vol. 1663, pp. 157–168. Springer, Heidelberg (1999)
26. Liu, Z., Parthasarathy, S., Ranganathan, A., Yang, H.: Near-optimal algorithms for shared filter evaluation in data stream systems. In: SIGMOD, pp. 133–146 (2008)
27. Munagala, K., Srivastava, U., Widom, J.: Optimization of continuous queries with shared expensive filters. In: PODS, pp. 215–224 (2007)
28. Nagarajan, V.: Approximation Algorithms for Sequencing Problems. PhD thesis, Tepper School of Business, Carnegie Mellon University (2009)
29. Nowak, R.D.: The geometry of generalized binary search. arXiv.org:0910.4397 (2009)
30. Schalekamp, F., Shmoys, D.: Algorithms for the universal and a priori TSP. Operations Research Letters 36(1), 1–3 (2008)
31. Shmoys, D., Talwar, K.: A Constant Approximation Algorithm for the a priori Traveling Salesman Problem. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 331–343. Springer, Heidelberg (2008)