Learning-Augmented Query Policies for Minimum Spanning Tree with Uncertainty

Thomas Erlebach
Department of Computer Science, Durham University, UK

Murilo Santos de Lima
München, Germany

Nicole Megow
Faculty of Mathematics and Computer Science, University of Bremen, Germany

Jens Schlöter
Faculty of Mathematics and Computer Science, University of Bremen, Germany

Abstract

We study how to utilize (possibly erroneous) predictions in a model for computing under uncertainty in which an algorithm can query unknown data. Our aim is to minimize the number of queries needed to solve the minimum spanning tree problem, a fundamental combinatorial optimization problem that has been central also to the research area of explorable uncertainty. For all integral $\gamma \geq 2$, we present algorithms that are γ-robust and $(1+\frac{1}{\gamma})$-consistent, meaning that they use at most γOPT queries if the predictions are arbitrarily wrong and at most $(1+\frac{1}{\gamma})$OPT queries if the predictions are correct, where OPT is the optimal number of queries for the given instance. Moreover, we show that this trade-off is best possible. Furthermore, we argue that a suitably defined *hop distance* is a useful measure for the amount of prediction error and design algorithms with performance guarantees that degrade smoothly with the hop distance. We also show that the predictions are PAC-learnable in our model. Our results demonstrate that untrusted predictions can circumvent the known lower bound of 2, without any degradation of the worst-case ratio. To obtain our results, we provide new structural insights for the minimum spanning tree problem that might be useful in the context of query-based algorithms regardless of predictions. In particular, we generalize the concept of witness sets – the key to lower-bounding the optimum – by proposing novel global witness set structures and completely new ways of adaptively using those.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases explorable uncertainty, queries, untrusted predictions

Digital Object Identifier 10.4230/LIPIcs.ESA.2022.49

Related Version *Full Version*: https://arxiv.org/abs/2206.15201

Funding *Thomas Erlebach*: Supported by EPSRC grant EP/S033483/2.
Murilo Santos de Lima: Work done while employed at University of Leicester, United Kingdom, and funded by EPSRC grant EP/S033483/1.
Nicole Megow: Supported by the German Science Foundation (DFG) under contract ME 3825/1.
Jens Schlöter: Funded by the German Science Foundation (DFG) under contract ME 3825/1.

1 Introduction

We introduce learning-augmented algorithms to the area of optimization under explorable uncertainty and focus on the fundamental *minimum spanning tree (MST) problem under explorable uncertainty*. We are given a (multi)graph $G = (V, E)$ with unknown edge weights $w_e \in \mathbb{R}_+$, for $e \in E$. For each edge e, an uncertainty interval I_e is known that contains w_e. A *query* on edge e reveals the true value w_e. The task is to determine an MST, i.e., a tree that connects all vertices of G, of minimum total weight w.r.t. the true values w_e. A query set is called *feasible* if it reveals sufficient information to identify an MST (not necessarily its exact weight). As queries are costly, the goal is to find a feasible query set of minimum size.

We study *adaptive strategies* that make queries sequentially and utilize the results of previous steps to decide upon the next query. As there exist input instances that are impossible to solve without querying all edges, we evaluate our algorithms in an *instance-dependent* manner: For each input, we compare the number of queries made by an algorithm with the best possible number of queries *for that input*, using competitive analysis. For a given problem instance, let OPT denote an arbitrary optimal query set (we later give a formal definition of OPT). An algorithm is *ρ-competitive* if it executes, for any problem instance, at most $\rho \cdot |\mathrm{OPT}|$ queries. While MST under explorable uncertainty is not a classical online problem where the input is revealed passively over time, the query results are uncertain and, to a large degree, dictate whether decisions to query certain edges were good or not. For analyzing an algorithm, it is natural to assume that the query results are determined by an adversary.This gives the problem a clear online flavor and prohibits the existence of 1-competitive algorithms even if we have unlimited running time and space [24]. We note that competitive algorithms in general do not have any running time requirements, but all our algorithm run in polynomial time.

The MST problem is among the most widely studied problems in the research area of *explorable uncertainty* [35] and has been a cornerstone in the development of algorithmic approaches and lower bound techniques [21, 22, 24, 27, 43, 44]. The best known deterministic algorithm for MST with uncertainty is 2-competitive, and no deterministic algorithm can be better [24]. A randomized algorithm with competitive ratio 1.707 is known [43]. Further work considers the non-adaptive problem, which has a very different flavor [44].

In this paper, we assume that an algorithm has, for each edge e, access to a prediction $\overline{w}_e \in I_e$ for the unknown value w_e. For example, machine learning (ML) methods could be used to predict the value of an edge. Given the tremendous progress in artificial intelligence and ML in recent decades, we can expect that those predictions are of good accuracy, but there is no guarantee and the predictions might be completely wrong. This lack of provable performance guarantees for ML often causes concerns regarding how confident one can be that an ML algorithm will perform sufficiently well in all circumstances. We address the very natural question whether the availability of such (ML) predictions can be exploited by query algorithms for computing with explorable uncertainty. Ideally, an algorithm should perform very well if predictions are accurate, but even if they are arbitrarily wrong, the algorithm should not perform worse than an algorithm without access to predictions. To emphasize that the predictions can be wrong, we refer to them as *untrusted predictions*.

We note that the availability of both uncertainty intervals and untrusted predictions is natural in many scenarios. For example, the quality of links (measured using metrics such as throughput and reliability) in a wireless network often fluctuates over time within a certain interval, and ML methods can be used to predict the precise link quality based on time-series data of previous link quality measurements [1]. The actual quality of a link can be obtained via a new measurement. If we wish to build a minimum spanning tree using links that currently have the highest link quality and want to minimize the additional measurements needed, we arrive at an MST problem with uncertainty and untrusted predictions.

We study for the first time the combination of explorable uncertainty and untrusted predictions. Our work is inspired by the vibrant recent research trend of considering untrusted (ML) predictions in the context of *online* algorithms, a different uncertainty model where the input is revealed to an algorithm incrementally. Initial work on online caching problems [40] has initiated a vast growing line of research on caching [5, 49, 53], rent-or-buy problems [31, 48, 54], scheduling [4, 9, 38, 45, 48], graph problems [19, 37, 39] and many more.

We adopt the following notions introduced in [40, 48]: An algorithm is α-*consistent* if it is α-competitive when the predictions are correct, and it is β-*robust* if it is β-competitive no matter how wrong the predictions are. Furthermore, we are interested in a smooth transition between the case with correct predictions and the case with arbitrarily incorrect predictions. We aim for performance guarantees that degrade gracefully with increasing prediction error.

Given predicted values for the uncertainty intervals, it is tempting to simply run an optimal algorithm under the assumption that the predictions are correct. This is obviously optimal with respect to consistency, but might give arbitrarily bad solutions in the case when the predictions are faulty. Instead of blindly trusting the predictions, we need more sophisticated strategies to be robust against prediction errors. This requires new lower bounds on an optimal solution, new structural insights, and new algorithmic techniques.

Main results

In this work, we show that, in the setting of explorable uncertainty, it is in fact possible to exploit ML predictions of the uncertain values and improve the performance of a query strategy when the predictions are good, while at the same time guaranteeing a strong bound on the worst-case performance even when the predictions are arbitrarily bad.

We give algorithms for the MST problem with uncertainty that are parameterized by a hyperparameter γ that reflects the user's confidence in the accuracy of the predictor. For any integral $\gamma \geq 2$, we present a $(1 + \frac{1}{\gamma})$-consistent and γ-robust algorithm, and show that this is the best possible trade-off between consistency and robustness. In particular, for $\gamma = 2$, we obtain a 2-robust, 1.5-consistent algorithm. It is worth noting that this algorithm achieves the improved competitive ratio of 1.5 for accurate predictions while maintaining the worst-case ratio of 2. This is in contrast to many learning-augmented online algorithms where the exploitation of predictions usually incurs an increase in the worst-case ratio (e.g., [6, 48]).

Our main result is a second and different algorithm with a more fine-grained performance analysis that obtains a guarantee that improves with the accuracy of the predictions. Very natural, simple error measures such as the number of inaccurate predictions or the ℓ_1-norm of the difference between predictions and true values turn out to prohibit any reasonable error-dependency. Therefore, we propose an error measure, called *hop distance* k_h, that takes structural insights about uncertainty intervals into account and may also be useful for other problems in computing with uncertainty and untrusted predictions. We give a precise definition of this error measure later. We also show in the full version [20] that the predictions are efficiently PAC-learnable with respect to k_h. Our main result is a learning-augmented algorithm with a competitive ratio with a linear error-dependency $\min\{(1+\frac{1}{\gamma})+\frac{5\cdot k_h}{|\mathrm{OPT}|}, \gamma+1\}$, for any integral $\gamma \geq 2$. All our algorithms have polynomial running-times. We describe our techniques and highlight their novelty in the following section.

The integrality requirement for γ comes from using γ to determine set sizes and can be removed by randomization at the cost of a slightly worse consistency guarantee; for a proof we refer to the full version.

Further related work

There is a long history of research on the tradeoff between exploration and exploitation when coping with uncertainty in the input data. Often, stochastic models are assumed, e.g., in work on multi-armed bandits [16, 28, 52], Weitzman's Pandora's box [55], and query-variants of combinatorial optimization problems; see, e.g., [32, 41, 51] and many more. In our work, we assume no knowledge of stochastic information and aim for robust algorithms that perform well even in a worst case.

The line of research on explorable uncertainty has been initiated by Kahan [35] in the context of selection problems. Subsequent work addressed finding the k-th smallest value in a set of uncertainty intervals [26, 33], caching problems [47], computing a function value [36], sorting [34], and classical combinatorial optimization problems. Some of the major aforementioned results on the MST problem under explorable uncertainty have been extended to general matroids [23, 43, 44]. Further problems that have been studied are the shortest path problem [25], the knapsack problem [29] and scheduling problems [2, 3, 7, 10, 18]. Although optimization under explorable uncertainty has been studied mainly in an adversarial model, recently first results have been obtained for stochastic variants for sorting [17] and selection type problems (hypergraph orientation) [11].

There is a significant body of work on computing in models where information about a hidden object can be accessed only via queries. The hidden object can, for example, be a function, a matrix, or a graph. In the graph context, property testing [30] has been studied extensively and there are many more works, see [8, 12, 13, 42, 46, 50]. The bounds on the number of queries made by an algorithm are typically absolute (as a function of the input) and the resulting correctness guarantees are probabilistic. This is very different from our work, where we aim for a comparison to the minimum number of queries needed for the given graph.

2 Overview of techniques and definition of error measure

We assume that each uncertainty interval is either *open*, $I_e = (L_e, U_e)$, or *trivial*, $I_e = \{w_e\}$, and we refer to edge e as *non-trivial* or *trivial*, respectively; a standard assumption to avoid a simple lower bound of $|E|$ on the competitive ratio [24, 33].

Before we give an overview of the used techniques, we formally define feasible and optimal query sets. We say that a query set $Q \subseteq E$ is *feasible* if there exists a set of edges $T \subseteq E$ such that T is an MST for the true values w_e of all $e \in Q$ and *every possible* combination of edge weights in I_e for the unqueried edges $e \in E \setminus Q$. That is, querying a feasible query set Q must give us sufficient information to identify a spanning tree T that is an MST for the true values no matter what the true values of the unqueried edges $E \setminus Q$ actually are. We call a feasible query set Q *optimal* if it has minimum cardinality $|Q|$ among all feasible query sets. Thus, the optimal solution depends only on the true values and not on the predicted values.

As Erlebach and Hoffmann [21] give a polynomial-time algorithm that computes an optimal query set under the assumption that all query results are known upfront, we can use their algorithm to compute the optimal query set under the assumption that all predicted values match the actual edge weights and query the computed set in an arbitrary order. If the predicted values are indeed correct, this yields a 1-consistent algorithm. However, such an algorithm may have an arbitrarily bad performance if the predictions are incorrect. Similarly, the known deterministic 2-competitive algorithm for the MST problem with uncertainty (without predictions) [24] is 2-robust and 2-consistent. The known lower bound of 2 rules out any robustness factor less than 2. It builds on the following simple example with two intersecting intervals I_a, I_b that are candidates for the largest edge weight in a cycle. No matter which interval a deterministic algorithm queries first, say I_a, the realized value could be $w_a \in I_a \cap I_b$, which requires a second query. If the adversary chooses $w_b \notin I_a \cap I_b$, querying just I_b would have been sufficient to identify the interval with larger true value. See also [24, Example 3.8] and [43, Section 3] for an illustration of the lower bound example.

Algorithm overview

We aim for $(1+\frac{1}{\gamma})$-consistent and γ-robust algorithms for each integral $\gamma \geq 2$. The algorithm proceeds in two phases: The first phase runs as long as there are prediction mandatory edges, i.e., edges that must be contained in every feasible query set under the assumption that the predictions are correct; we later give a formal characterization of such edges. In this phase, we exploit the existence of those edges and their properties to execute queries with strong local guarantees, i.e., each feasible query set contains a large portion of our queries. For the second phase, we observe and exploit that the absence of prediction mandatory queries implies that the predicted optimal solution is a minimum vertex cover in a bipartite auxiliary graph. The challenge here is that the auxiliary graph can change with each wrong prediction. To obtain an error-dependent guarantee (our error measure k_h is discussed below) we need to adaptively query a dynamically changing minimum vertex cover.

Novel techniques

During the first phase, we generalize the classical witness set analysis. In computing with explorable uncertainty, the concept of *witness sets* is crucial for comparing the query set of an algorithm with an optimal solution (a way of lower-bounding). A witness set [15] is a set of elements for which we can guarantee that any feasible solution must query at least *one* of these elements. Known algorithms for the MST problem without predictions [24, 43] essentially follow the algorithms of Kruskal or Prim and only identify witness sets of size 2 in the cycle or cut that is currently under consideration. Querying disjoint witness sets of size 2 (witness pairs) ensures 2-robustness but does not lead to an improved consistency.

In our first phase, we extend this concept by considering *strengthened* witness sets of three elements such that any feasible query set must contain at least two of them. Since we cannot always find strengthened witness sets based on structural properties alone (otherwise, there would be a 1.5-competitive algorithm for the problem without predictions), we identify such sets under the assumption that the predictions are correct. Even after identifying such elements, the algorithm needs to query them in a careful order: if the predictions are wrong, we lose the guarantee on the elements, and querying all of them might violate the robustness. In order to identify strengthened witness sets, we provide new, more global criteria to identify additional witness sets (of size two) beyond the current cycle or cut. These new ways to identify witness sets are a major contribution that may be of independent interest regardless of predictions. During the first phase, we repeatedly query $\gamma - 2$ prediction mandatory edges together with a strengthened witness set, which ensures $(1 + \frac{1}{\gamma})$-consistency. We query the elements in a carefully selected order while adjusting for errors to ensure γ-robustness.

For the second phase, we observe that the predicted optimal solution of the remaining instance is a minimum vertex cover VC in a bipartite auxiliary graph representing the structure of *potential* witness pairs (edges of the input graph correspond to vertices of the auxiliary graph). For instances with this property, we aim for 1-consistency and 2-robustness; the best-possible trade-off for such instances. If the predictions are correct, each edge of the auxiliary graph is a witness pair. However, if a prediction error is observed when a vertex of VC is queried, the auxiliary graph changes. This means that some edges of the original auxiliary graph are not actually witness pairs. Indeed, the size of a minimum vertex cover can increase and decrease and does not constitute a lower bound on $|\mathrm{OPT}|$; we refer to the full version for an example.

If we only aim for consistency and robustness, we can circumvent this problem by selecting a distinct matching partner $h(e) \notin VC$ for each $e \in VC$ applying *Kőnig-Egerváry*'s Theorem (duality of maximum matchings and minimum vertex covers in bipartite graphs, see e.g. [14]).

By querying the elements of VC in a carefully chosen order until a prediction error is observed for the first time, we can guarantee that $\{e, h(e)\}$ is a witness set for each $e \in VC$ that is already queried. In the case of an error, this allows us to extend the previously queried elements to disjoint witness pairs to guarantee 2-robustness. Then, we can switch to an arbitrary (prediction-oblivious) 2-competitive algorithm for the remaining queries.

If we additionally aim for an error-sensitive guarantee, however, handling the dynamic changes to the auxiliary graph, its minimum vertex cover VC and matching h requires substantial additional work, and we see overcoming this challenge as our main contribution. In particular, querying the partner $h(e)$ of each already queried $e \in VC$ in case of an error might be too expensive for the error-dependent guarantee. However, if we do not query these partners, the changed instance still depends on them, and if we charge against such a partner multiple times, we can lose the robustness. Our solution is based on an elaborate charging/counting scheme and involves:

- keeping track of matching partners of already queried elements of VC;
- updating the matching and VC using an augmenting path method to bound the number of elements that are charged against multiple times in relation to the prediction error;
- and querying the partners of previously queried edges (and their new matching partners) as soon as they become endpoints of a newly matched edge, in order to prevent dependencies between the (only partially queried) witness sets of previously queried edges.

The error-sensitive algorithm achieves a competitive ratio of $1 + \frac{1}{\gamma} + \frac{5k_h}{|\mathrm{OPT}|}$, at the price of a slightly increased robustness of $\gamma + 1$ instead of γ.

Hop distance as error metric

When we aim for a fine-grained performance analysis giving guarantees that depend on the quality of predictions, we need a metric to measure the prediction error. A very natural, simple error measure is the number of inaccurate predictions $k_\# = |\{e \in E \,|\, w_e \neq \overline{w}_e\}|$. However, we can show that even for $k_\# = 1$ the competitive ratio cannot be better than the known bound of 2 (see Lemma 19 in the full version). The reason for the weakness of this measure is that it completely ignores the interleaving structure of intervals. Similarly, an ℓ_1 error metric such as $\sum_{e \in E} |w_e - \overline{w}_e|$ would not be meaningful because only the *order* of the values and the interval endpoints matters for our problems.

We propose a refined error measure, which we call *hop distance*. It is very intuitive even though it requires some technical care to make it precise. If we consider only a single predicted value $\overline{w}_e$ for some $e \in E$, then, in a sense, this value predicts the relation of the true value w_e to the intervals of edges $e' \in E \setminus \{e\}$. In particular, w.r.t. a fixed $e' \in E \setminus \{e\}$, the value $\overline{w}_e$ predicts whether w_e is left of $I_{e'}$ ($\overline{w}_e \leq L_{e'}$), right of $I_{e'}$ ($\overline{w}_e \geq U_{e'}$), or contained in $I_{e'}$ ($L_{e'} < \overline{w}_e < U_{e'}$). Interpreting the prediction $\overline{w}_e$ in this way, the prediction is "wrong" (w.r.t. a fixed $e' \in E \setminus \{e\}$) if the predicted relation of the true value w_e to interval $I_{e'}$ is not actually true, e.g., w_e is predicted to be left of $I_{e'}$ ($\overline{w}_e \leq L_{e'}$) but the actual w_e is either contained in or right of $I_{e'}$ ($w_e > L_{e'}$). Formally, we define the function $k_{e'}(e)$ that indicates whether the predicted relation of w_e to $I_{e'}$ is true ($k_{e'}(e) = 0$) or not ($k_{e'}(e) = 1$). With the prediction error $k^+(e)$ for a single $e \in E$, we want to capture the number of relations between w_e and intervals $I_{e'}$ with $e' \in E \setminus \{e\}$ that are not accurately predicted. Thus, we define $k^+(e) = \sum_{e' \in E \setminus \{e\}} k_{e'}(e)$. For a set of edges $E' \subseteq E$, we define $k^+(E') = \sum_{e \in E'} k^+(e)$. Consequently, with the error for the complete instance we want to capture the total number of wrongly predicted relations and, therefore, define it by $k_h = k^+(E)$. We call this error measure k_h the *hop distance*; see Figure 1 for an illustration.

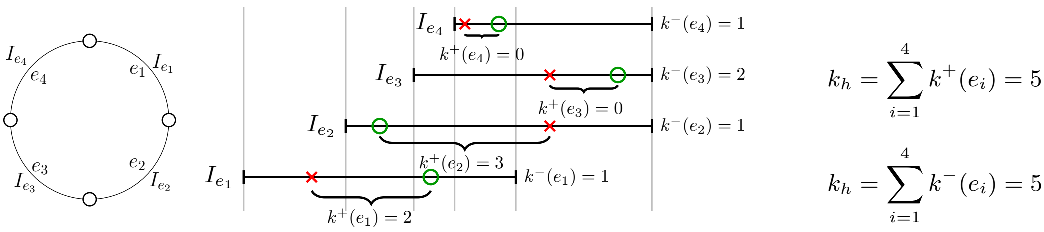

Figure 1 Example of a single cycle (left) with uncertain edge weights from intersecting intervals $I_{e_1}, I_{e_2}, I_{e_3}, I_{e_4}$ (middle). Circles illustrate true values and crosses illustrate the predicted values.

Symmetrically, we can define $k^-(e) = \sum_{e' \in E \setminus \{e\}} k_e(e')$ and $k^-(E') = \sum_{e \in E'} k^-(e)$ for subset $E' \subseteq E$. Then $k^+(E) = k_h = k^-(E)$ follows by reordering the summations.

3 Structural results

In this section, we introduce some known concepts and prove new structural results on MST under explorable uncertainty, which we use later to design learning-augmented algorithms.

Witness sets and mandatory queries

Witness sets are the key to the analysis of query algorithms. They allow for a comparison of an algorithm's query set to an optimal solution. A subset $W \subseteq E$ is a *witness set* if $W \cap Q \neq \emptyset$ for all feasible query sets Q. An important special case are witness sets of cardinality one, i.e., edges that are part of every feasible query set. We call such edges *mandatory*. Similarly, we call edges that are mandatory under the assumption that the predictions are correct *prediction mandatory*.

For an example, consider Figure 2. In the example, we see the uncertainty intervals, predicted values and true values of four edges that form a simple cycle. We can observe that both e_1 and e_2 are mandatory for this example. To see this, assume that e_1 is not mandatory. Then, there must be a feasible query set Q with $e_1 \notin Q$ for the instance, which implies that $Q = \{e_2, e_3, e_4\}$ must be feasible. But even after querying Q to reveal the true values of e_2, e_3 and e_4, it still depends on the still unknown true value of e_1 whether there exists an MST T with $e_1 \in T$ (only if $w_{e_1} \leq w_{e_2}$) and/or $e_1 \notin T$ (only if $w_{e_2} \leq w_{e_1}$). Even after querying Q there is no spanning tree T that is an MST for each possible edge weight in I_{e_1} of the unqueried edge e_1 and, thus, Q is not feasible. This implies that e_1 is mandatory, and we can argue analogously that e_2 is mandatory as well.

To argue whether an edge is prediction mandatory, on the other hand, we assume that all queries reveal the predicted values as true values. Under this assumption, a query to e_1 in the example would reveal a value that is larger than all upper limits U_{e_i} with $i \in \{2, 3, 4\}$, which implies that e_1 cannot be part of any MST and that $T = \{e_2, e_3, e_4\}$ is an MST no matter what the true values of e_2, e_3 and e_4 actually are. Therefore, under the assumption that all predicted values match the true values, $Q = \{e_1\}$ is a feasible query set and, thus, e_2 is not prediction mandatory despite being mandatory. However, we can use a similar argumentation as above to argue that e_1 is also prediction mandatory.

We continue by giving properties that allow us to identify (prediction) mandatory edges. To that end, let the *lower limit tree* $T_L \subseteq E$ be an MST for values w^L with $w^L_e = L_e + \epsilon$ for an infinitesimally small $\epsilon > 0$. Analogously, let the *upper limit tree* T_U be an MST for values w^U with $w^U_e = U_e - \epsilon$. This concept has been introduced in [43] to identify mandatory

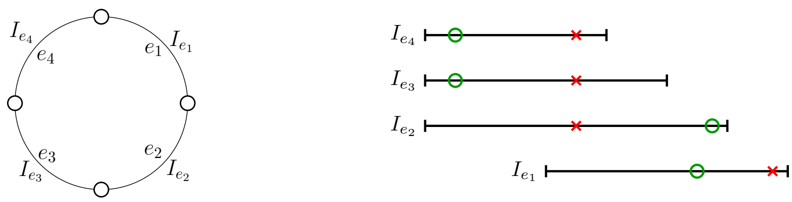

Figure 2 Example of a single cycle (left) with uncertain edge weights from intersecting intervals $I_{e_1}, I_{e_2}, I_{e_3}, I_{e_4}$ (right). Circles illustrate true values and crosses illustrate the predicted values. For the example, $\{e_1, e_2\}$ is the set of all mandatory edges and $\{e_1\}$ is the set of all prediction mandatory edges.

queries; it is shown that every non-trivial $e \in T_L \setminus T_U$ is mandatory. Thus, we may repeatedly query edges in $T_L \setminus T_U$ until $T_L = T_U$ without worsening the competitive ratio. We can extend this preprocessing to achieve uniqueness for T_L and T_U and, thus, may assume unique $T_L = T_U$.

▶ **Lemma 1.** *By querying only mandatory elements we can obtain an instance with $T_L = T_U$ such that T_L and T_U are the unique lower limit tree and upper limit tree, respectively.*

Given an instance with unique $T_L = T_U$, we observe that each $e \in T_L$ that is not part of the MST for the true values is mandatory. Similarly, each $e \notin T_L$ that is part of the MST for the true values is mandatory as well. A stronger version of this observation is as follows.

▶ **Lemma 2.** *Let G be an instance with unique $T_L = T_U$ and let G' be an instance with unique $T'_L = T'_U$ obtained from G by querying set Q, then $e \in T_L \Delta T'_L = (T_L \setminus T'_L) \cup (T'_L \setminus T_L)$ implies $e \in Q$.*

Next we establish a relation between the set $E_M \subseteq E$ of mandatory queries, the set $E_P \subseteq E$ of prediction mandatory queries, and the hop distance k_h.

▶ **Lemma 3.** *Consider a given problem instance $G = (V, E)$ with predicted values $\overline{w}$. Each $e \in E_M \Delta E_P$ satisfies $k^-(e) \geq 1$. Consequently, $k_h \geq |E_M \Delta E_P|$.*

For a formal proof, we refer to the full version. Intuitively, if $e \in E_P \setminus E_M$, then it is possible to solve the instance without querying e. Thus, the relation of the true values $w_{e'}$ with $e' \in E \setminus \{e\}$ to interval I_e must be such that querying $E \setminus \{e\}$ allows us to verify that e is either part or not part of the MST. If the predicted relations of the true values $w_{e'}$ with $e' \in E \setminus \{e\}$ to interval I_e were the same, then querying $E \setminus \{e\}$ would also allow us to verify that e is either part or not part of the MST. Thus, the predicted relation of at least one $w_{e'}$ to interval I_e must be wrong. We can argue analogously for $e \in E_M \setminus E_P$.

Identifying witness sets

We introduce new structural properties to identify witness sets. Existing algorithms for MST under uncertainty [24, 43] essentially follow the algorithms of Kruskal or Prim, and only identify witness sets in the cycle or cut that is currently under consideration. Let $f_1, \ldots, f_l$ denote the edges in $E \setminus T_L$ ordered by non-decreasing lower limit. Then, C_i with $i \in \{1, \ldots, l\}$ denotes the unique cycle in $T_L \cup \{f_i\}$.

For each $e \in T_L$, let X_e denote the set of edges in the cut of the two connected components of $T_L \setminus \{e\}$. Existing algorithms for MST under explorable uncertainty repeatedly consider (the changing) C_1 or X_e, where e is the edge in T_L with maximum upper limit, and identify the maximum or minimum edge in the cycle or cut by querying witness sets of size two, until the problem is solved. For our algorithms, we need to identify witness sets in cycles $C_i \neq C_1$ and cuts $X_{e'} \neq X_e$.

▶ **Lemma 4.** *Consider cycle C_i with $i \in \{1, \ldots, l\}$. Let $l_i \in C_i \setminus \{f_i\}$ such that $I_{l_i} \cap I_{f_i} \neq \emptyset$ and l_i has the largest upper limit in $C_i \setminus \{f_i\}$, then $\{f_i, l_i\}$ is a witness set. If $w_{f_i} \in I_{l_i}$, then l_i is mandatory.*

Characterization of prediction mandatory free instances

We say an instance is *prediction mandatory free* if it contains no prediction mandatory elements. A key part of our algorithms is to transform instances into prediction mandatory free instances while maintaining a competitive ratio that allows us to achieve the optimal consistency and robustness trade-off overall. We give the following characterization of prediction mandatory free instances, (cf. Figure 3). Then, we show that prediction mandatory free instances remain so as long as we ensure $T_L = T_U$.

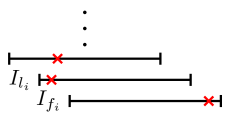

Figure 3 Intervals in a prediction mandatory free cycle with predictions indicated as red crosses.

▶ **Lemma 5.** *An instance G is prediction mandatory free if and only if $\overline{w}_{f_i} \geq U_e$ and $\overline{w}_e \leq L_{f_i}$ holds for each $e \in C_i \setminus \{f_i\}$ and each cycle C_i with $i \in \{1, \ldots, l\}$. Once an instance is prediction mandatory free, it remains so even if we query further elements, as long as we maintain unique $T_L = T_U$.*

Making instances prediction mandatory free

In the full version, we give a powerful preprocessing algorithm that transforms arbitrary instances into prediction mandatory free instances.

▶ **Theorem 6.** *There is an algorithm that makes a given instance prediction mandatory free and satisfies $|\mathrm{ALG}| \leq \min\{(1 + \frac{1}{\gamma}) \cdot (|(\mathrm{ALG} \cup D) \cap \mathrm{OPT}| + k^+(\mathrm{ALG}) + k^-(\mathrm{ALG})), \gamma \cdot |(\mathrm{ALG} \cup D) \cap \mathrm{OPT}| + \gamma - 2\}$ for the set of edges ALG queried by the algorithm and a set $D \subseteq E \setminus \mathrm{ALG}$ of unqueried edges that do not occur in the remaining instance* after *executing the algorithm.*

The set D are edges that, even without being queried by the algorithm, are proven to be maximal in a cycle or minimal in a cut. Thus, they can be deleted or contracted w.l.o.g. and do not exist in the instance remaining *after* executing the preprocessing algorithm. This is an important property as it means that the remaining instance is independent of D and ALG (as all elements of ALG are already queried). Since the theorem compares $|\mathrm{ALG}|$ with $|(\mathrm{ALG} \cup D) \cap \mathrm{OPT}|$ instead of just $|\mathrm{OPT}|$, this allows us to combine the given guarantee with the guarantees of dedicated algorithms for prediction mandatory free instances. However, we have to be careful with the additive term $\gamma - 2$, but we will see that we can charge this term against the improved robustness of our algorithms for prediction mandatory free instances.

4 An algorithm with optimal consistency and robustness trade-off

We give a bound on the best achievable tradeoff between consistency and robustness.

▶ **Theorem 7.** *Let $\beta \geq 2$ be a fixed integer. For the MST problem under explorable uncertainty with predictions, there is no deterministic β-robust algorithm that is α-consistent for $\alpha < 1 + \frac{1}{\beta}$. And vice versa, no deterministic α-consistent algorithm, with $\alpha > 1$, is β-robust for $\beta < \max\{\frac{1}{\alpha-1}, 2\}$.*

The main result of this section is an optimal algorithm w.r.t. this tradeoff bound.

▶ **Theorem 8.** *For every integer $\gamma \geq 2$, there exists a $(1 + \frac{1}{\gamma})$-consistent and γ-robust algorithm for the MST problem under explorable uncertainty with predictions.*

To show this result, we design an algorithm for prediction mandatory free instances with unique $T_L = T_U$. We run it after the preprocessing algorithm which obtains such special instance with the query guarantee in Theorem 6. Our new algorithm achieves the optimal trade-off.

▶ **Theorem 9.** *There exists a 1-consistent and 2-robust algorithm for prediction mandatory free instances with unique $T_L = T_U$ of the MST problem under explorable uncertainty with predictions.*

In a prediction mandatory free instance $G = (V, E)$, each $f_i \in E \setminus T_L$ is predicted to be maximal on cycle C_i, and each $l \in T_L$ is predicted to be minimal in X_l (cf. Lemma 5). If these predictions are correct, then T_L is an MST and the optimal query set is a minimum vertex cover in a bipartite graph $\bar{G} = (\bar{V}, \bar{E})$ with $\bar{V} = E$ (excluding trivial edges) and $\bar{E} = \{\{f_i, e\} \mid i \in \{1, \ldots, l\}, e \in C_i \setminus \{f_i\} \text{ and } I_e \cap I_{f_i} \neq \emptyset\}$ [21, 43]. We refer to $\bar{G}$ as the *vertex cover instance*. Note that if a query reveals that an f_i is not maximal on C_i or an $l \in T_L$ is not minimal in X_l, then the vertex cover instance changes. Let VC be a minimum vertex cover of $\bar{G}$. Non-adaptively querying VC ensures 1-consistency but might lead to an arbitrarily bad robustness. Indeed, the size of a minimum vertex cover can increase and decrease drastically as we show in the full version. Thus, the algorithm has to be more adaptive.

The idea of the algorithm (cf. Algorithm 1) is to sequentially query each $e \in VC$ and charge for querying e by a distinct non-queried element $h(e)$ such that $\{e, h(e)\}$ is a witness set. Querying exactly one element per distinct witness set implies optimality. To identify $h(e)$ for each element $e \in VC$, we use the fact that *Kőnig-Egerváry*'s Theorem (e.g., [14]) on the duality between minimum vertex covers and maximum matchings in bipartite graphs implies that there is a matching h that maps each $e \in VC$ to a distinct $e' \notin VC$. While the sets $\{e, h(e)\}$ with $e \in VC$ in general are not witness sets, querying VC in a specific order until the vertex cover instance changes guarantees that $\{e, h(e)\}$ is a witness set for each already queried e. The algorithm queries in this order until it detects a wrong prediction or solves the problem. If it finds a wrong prediction, it queries the partner $h(e)$ of each already queried edge e, and continues to solve the problem with a 2-competitive algorithm (e.g., [24, 43]). The following lemma specifies the order in which the algorithm queries VC.

▶ **Lemma 10.** *Let $l'_1, \ldots, l'_k$ be the edges in $VC \cap T_L$ ordered by non-increasing upper limit and let d be such that the true value of each l'_i with $i < d$ is minimal in cut $X_{l'_i}$, then $\{l'_i, h(l'_i)\}$ is a witness set for each $i \leq d$. Let $f'_1, \ldots, f'_g$ be the edges in $VC \setminus T_L$ ordered by non-decreasing lower limit and let b be such that the true value of each f'_i with $i < b$ is maximal in cycle $C_{f'_i}$, then $\{f'_i, h(f'_i)\}$ is a witness set for each $i \leq b$.*

Algorithm 1 1-consistent and 2-robust algorithm for prediction mandatory free instances.

Input: Prediction mandatory free graph $G = (V, E)$ with unique $T_L = T_U$.
1 Compute maximum matching h and minimum vertex cover VC for $\bar{G}$;
2 Set $W = \emptyset$, and let $f'_1, \ldots, f'_g$ and $l'_1, \ldots, l'_k$ be as described in Lemma 10;
3 **for** *e chosen sequentially from the ordered list $f'_1, \ldots, f'_g, l'_1, \ldots, l'_k$* **do**
4   Query e and add $h(e)$ to W;
5   **if** $k^+(e) \neq 0$ **then** query set W and solve the instance with a 2-competitive algorithm;

Proof. Here, we show the first statement and refer to the full version for the proof of the second statement. Consider an arbitrary l'_i and $h(l'_i)$ with $i \leq d$. By definition of h, the edge $h(l'_i)$ is not part of the lower limit tree. Consider $C_{h(l'_i)}$, i.e., the cycle in $T_L \cup \{h(l'_i)\}$, then we claim that $C_{h(l'_i)}$ only contains $h(l'_i)$ and edges in $\{l'_1, \ldots, l'_k\}$ (and possibly irrelevant edges that do not intersect $I_{h(l'_i)}$). To see this, recall that $l'_i \in VC$, by definition of h, implies $h(l'_i) \notin VC$. For VC to be a vertex cover, each $e \in C_{h(l'_i)} \setminus \{h(l'_i)\}$ must either be in VC or not intersect $h(l'_i)$. Consider the relaxed instance where the true values for each l'_j with $j < d$ and $j \neq i$ are already known. By assumption each such l'_j is minimal in its cut $X_{l'_j}$. Thus, we can w.l.o.g. contract each such edge. It follows that in the relaxed instance l'_i has the highest upper limit in $C_{h(l'_i)} \setminus \{h(l'_i)\}$. According to Lemma 4, $\{l'_i, h(l'_i)\}$ is a witness set. ◀

Proof of Theorem 9. We first show 1-consistency. Assume that all predictions are correct, then VC is an optimal query set and $k^+(e) = 0$ holds for all $e \in E$. It follows that Line 5 never executes queries and the algorithm queries exactly VC. This implies 1-consistency.

Further, if the algorithm never queries in Line 5, then the consistency analysis implies 1-robustness. Suppose Line 5 executes queries. Let Q_1 denote the set of edges that are queried before the queries of Line 5 and let $Q_2 = \{h(e) \mid e \in Q_1\}$. Then Q_2 corresponds to the set W as queried in Line 5. By Lemma 10, each $\{e, h(e)\}$ with $e \in Q_1$ is a witness set. Further, the sets $\{e, h(e)\}$ are pairwise disjoint. Thus, $|Q_1 \cup Q_2| \leq 2 \cdot |\mathrm{OPT} \cap (Q_1 \cup Q_2)|$. Apart from $Q_1 \cup Q_2$, the algorithm queries a set Q_3 in Line 5 to solve the remaining instance with a 2-competitive algorithm. So, $|Q_3| \leq 2 \cdot |\mathrm{OPT} \setminus (Q_1 \cup Q_2)|$ and, adding up the inequalities, $|\mathrm{ALG}| \leq 2 \cdot |\mathrm{OPT}|$. ◀

A careful combination of Theorems 6 and 9 implies Theorem 8. Full proof in the full version.

5 An error-sensitive algorithm

In this section, we extend the algorithm of Section 4 to obtain error sensitivity. First, we note that $k_\# = 0$ implies $k_h = 0$, so Theorem 7 implies that no algorithm can simultaneously have competitive ratio better than $1 + \frac{1}{\beta}$ if $k_h = 0$ and β for arbitrary k_h. In addition, we can give the following lower bound on the competitive ratio as a function of k_h.

▶ **Theorem 11.** *Any deterministic algorithm for MST under explorable uncertainty with predictions has a competitive ratio $\rho \geq \min\{1 + \frac{k_h}{|\mathrm{OPT}|}, 2\}$, even for edge disjoint prediction mandatory free cycles.*

Again, we design an algorithm for prediction mandatory free instances with unique $T_L = T_U$ and use it in combination with the preprocessing algorithm (Theorem 6) to prove the following.

▶ **Theorem 12.** *For every integer $\gamma \geq 2$, there exists a* $\min\{1 + \frac{1}{\gamma} + \frac{5 \cdot k_h}{|\mathrm{OPT}|}, \gamma + 1\}$*-competitive algorithm for the MST problem under explorable uncertainty with predictions.*

We actually show a robustness of $\max\{3, \gamma + \frac{1}{|\mathrm{OPT}|}\}$ which might be smaller than $\gamma + 1$. Our algorithm for prediction mandatory free instances asymptotically matches the error-dependent guarantee of Theorem 11 at the cost of a slightly worse robustness.

▶ **Theorem 13.** *There exists a* $\min\{1 + \frac{5 \cdot k_h}{|\mathrm{OPT}|}, 3\}$*-competitive algorithm for prediction mandatory free instances with unique $T_L = T_U$ of the MST problem under explorable uncertainty with predictions.*

We follow the same strategy as before. However, Algorithm 1 just executes a 2-competitive algorithm once it detects an error. This is sufficient to achieve the optimal trade-off as we, if an error occurs, only have to guarantee 2-competitiveness. To obtain an error-sensitive guarantee however, we have to ensure both, $|\mathrm{ALG}| \leq 3 \cdot |\mathrm{OPT}|$ *and* $|\mathrm{ALG}| \leq \mathrm{OPT} + 5 \cdot k_h$ even if errors occur. Further, we might not be able to afford queries to the complete set W (Algorithm 1, Line 5) in the case of an error as this might violate $|\mathrm{ALG}| \leq \mathrm{OPT} + 5 \cdot k_h$.

We adjust the algorithm to query elements of $f'_1, \dots, f'_g$ and $l'_1, \dots, l'_k$ as described in Lemma 10 not only until an error occurs but until the vertex cover instance changes. That is, until some f_i that at the beginning of the iteration is not part of T_L becomes part of the lower limit tree, or some l_i that at the beginning of the iteration is part of T_L is not part of the lower limit tree anymore. Once the instance changes, we recompute both, the bipartite graph $\bar{G}$ as well as the matching h and minimum vertex cover VC for $\bar{G}$. Instead of querying the complete set W, we only query the elements of W that occur in the recomputed matching, as well as the new matching partners of those elements. And instead of switching to a 2-competitive algorithm, we restart the algorithm with the recomputed matching and vertex cover. Algorithm 2 formalizes this approach. In the algorithm, h denotes a matching that matches each $e \in VC$ to a distinct $h(e) \notin VC$; we use the notation $\{e, e'\} \in h$ to indicate that h matches e and e'. For a subset $U \subseteq VC$ let $h(U) = \{h(e) \mid e \in U\}$. For technical reasons, the algorithm does not recompute an arbitrary matching h but follows the approach of Lines 10 and 11. Intuitively, an arbitrary maximum matching h might contain too many elements of W, which would lead to too many additional queries.

Let ALG denote the queries of Algorithm 2 on a prediction mandatory free instance with unique $T_L = T_U$. We show Theorem 13 by proving $\mathrm{ALG} \leq \mathrm{OPT} + 5 \cdot k_h$ and $\mathrm{ALG} \leq 3 \cdot \mathrm{OPT}$.

Before proving the two inequalities, we state some key observations about the algorithm. We argue that an element e' can never be part of a partial matching $\bar{h}$ in an execution of Line 10 *after* it is added to set W. Recall that the vertex cover instances only contain non-trivial elements. Thus, if an element e is queried in Line 5 and the current partner $e' = h(e)$ is added to set W, then the vertex cover instance at the next execution of Line 10 does not contain the edge $\{e, e'\}$ and, therefore, e' is not part of the partial matching $\bar{h}$ of that line. As long as e' is not added to the matching by Line 11, it, by definition, can never be part of a partial matching $\bar{h}$ in an execution of Line 10. As soon as the element e' is added to the matching in some execution of Line 11, it is queried in the following execution of Line 12. Therefore, e' can also not be part of a partial matching $\bar{h}$ in an execution of Line 10 after it is added to the matching again. This leads to the following observation.

▶ **Observation 14.** *An element e' can never be part of a partial matching $\bar{h}$ in an execution of Line 10* after *it is added to set W. Once e' is added to the matching again in an execution of Line 11, it is queried directly afterwards in Line 12, and cannot occur in Line 5 anymore.*

Algorithm 2 Error-sensitive algorithm for prediction mandatory free instances.

Input: Prediction mandatory free graph $G = (V, E)$ with unique $T_L = T_U$.
1 Compute maximum matching h and minimum vertex cover VC for $\bar{G}$ and set $W = \emptyset$;
2 Let $f'_1, \dots, f'_g$ and $l'_1, \dots, l'_k$ be as described in Lemma 10 for VC and h;
3 $L \leftarrow T_L$, $N \leftarrow E \setminus T_L$; `/* recompute the actual` T_L `after each query */`
4 **for** *e chosen sequentially from the ordered list $f'_1, \dots, f'_g, l'_1, \dots, l'_k$* **do**
5   If e is non-trivial, i.e., has not been queried yet, query e and add $h(e)$ to W;
6   Apply Lemma 1 to ensure unique $T_L = T_U$. For each query e', if $\exists a$ s.t. $\{e', a\} \in h$, query a ;
7   Let $\bar{G}' = (\bar{V}', \bar{E}')$ be the vertex cover instance for the current instance;
8   **if** *some $e' \in L$ is not in T_L or some $e' \in N$ is in T_L* **then**
9   **repeat**
10   Let $\bar{G} = \bar{G}'$ and $\bar{h} = \{\{e', e''\} \in h \mid \{e', e''\} \in \bar{E}'\}$;
11   Compute h and VC by completing $\bar{h}$ with an augmenting path algorithm;
12   Query $R = (VC \cup h(VC)) \cap (W \cup \{e' \mid \exists e \in W \text{ with } \{e, e'\} \in h\})$;
13   Ensure unique $T_L = T_U$. For each query e', if $\exists a$ s.t. $\{e', a\} \in h$, query a;
14   Let $\bar{G}' = (\bar{V}', \bar{E}')$ be the vertex cover instance for the current instance;
15   **until** $R = \emptyset$;
16   Restart at Line 2. In particular, do *not* reset W;

We first analyze the queries that are *not* executed in Line 12. Let $Q_1 \subseteq \mathrm{ALG}$ denote the queries of Line 5. For each $e \in Q_1$ let $h^*(e)$ be the matching partner of e at the time it was queried, and let $h^*(Q_1) = \bigcup_{e \in Q_1} \{h^*(e)\}$. Finally, let Q_2 denote the queries of Lines 6 and 13 to elements of $h^*(Q_1)$, and let Q_3 denote the remaining queries of those lines.

▶ **Lemma 15.** *$|Q_1 \cup Q_3 \cup h^*(Q_1)| \leq 2 \cdot |\mathrm{OPT} \cap (Q_1 \cup Q_3 \cup h^*(Q_1))|$ and $|Q_1 \cup Q_2 \cup Q_3| \leq |\mathrm{OPT} \cap (Q_1 \cup Q_3 \cup h^*(Q_1))| + k^-(Q_2 \cup Q_3)$.*

Proof. First, consider Q_1 and $h^*(Q_1)$. By Lemma 5, the instance is prediction mandatory free at the beginning of each restart of the algorithm. By Lemma 10, each $\{e, h^*(e)\}$ with $e \in Q_1$ is a witness set. We claim that all such $\{e, h^*(e)\}$ are pairwise disjoint, which implies $|Q_1 \cup h^*(Q_1)| \leq 2 \cdot |\mathrm{OPT} \cap (Q_1 \cup h^*(Q_1))|$. Otherwise, an element of $\{e, h^*(e)\}$ must occur a second time in Line 5 after e is queried and $h^*(e)$ is added to W. Thus, either e or $h^*(e)$ must become part of a recomputed matching in line 10. By Observation 14 and since e becomes trivial, this cannot happen.

Consider an $e \in Q_2 \subseteq h^*(Q_1)$ and let $e' \in Q_1$ with $h^*(e') = e$. Since $e' \in Q_1$, it was queried in Line 5. Observe that e must have been queried after e', as otherwise either e' would not have been queried in Line 5 (but together with e in Line 6 or 13), or e would not have been the matching partner of e' when it was queried by Observation 14; both contradict $e' \in Q_1$ and $h^*(e') = e$. This and Observation 14 imply that, at the time e is queried, its current matching partner is either the trivial e' or it has no partner. So, e must have been queried because it was mandatory and not as the matching partner of a mandatory element. Thus, each query of Q_2 is mandatory but, by Lemma 5, not prediction mandatory at the beginning of the iteration in which it is queried. Therefore, Lemma 3 implies that all mandatory elements e of Q_2 have $k^-(e) \geq 1$. It follows $|Q_1 \cup Q_2| \leq |\mathrm{OPT} \cap (Q_1 \cap h^*(Q_1))| + k^-(Q_2)$.

By the argument above, no element of Q_3 was queried as the matching partner to an element of $Q_2 \cup Q_1$. Thus, by Lemma 1 and the definition of the algorithm, at least half the elements of Q_3 are mandatory, and we have $|Q_3| \leq 2 \cdot |\mathrm{OPT} \cap Q_3|$ (and $\frac{1}{2}|Q_3| \leq |\mathrm{OPT} \cap Q_3|$), which implies $|Q_1 \cup Q_3 \cup h^*(Q_1)| \leq 2 \cdot |\mathrm{OPT} \cap (Q_1 \cup Q_3 \cup h^*(Q_1))|$.

By the same argument as for Q_2, all mandatory elements e of Q_3 have $k^-(e) \geq 1$. Thus, $k^-(Q_3) \geq \frac{1}{2} \cdot |Q_3|$. Combining $k^-(Q_3) \geq \frac{1}{2} \cdot |Q_3|$ and $\frac{1}{2}|Q_3| \leq |\mathrm{OPT} \cap Q_3|$ implies $|Q_3| \leq |\mathrm{OPT} \cap Q_3| + k^-(Q_3)$. So, $|Q_1 \cup Q_2 \cup Q_3| \leq |\mathrm{OPT} \cap (Q_1 \cup Q_3 \cup h^*(Q_1))| + k^-(Q_2 \cup Q_3)$. ◀

The first part of Lemma 15 captures all queries outside of Line 12 and all queries of Line 12 to elements of $W = h^*(Q_1)$. Let Q_4' be the remaining queries of Line 12. By definition of the algorithm, $|Q_4'| \leq |W|$. Since $|W| \leq |\mathrm{OPT}|$, we can conclude the next lemma.

▶ **Lemma 16.** $|\mathrm{ALG}| \leq 3 \cdot |\mathrm{OPT}|$.

Next, we show $|\mathrm{ALG}| \leq |\mathrm{OPT}| + 5 \cdot k_h$. Lemma 15 implies $|Q_1 \cup Q_2 \cup Q_3| \leq |\mathrm{OPT} \cap (Q_1 \cup Q_3 \cup h^*(Q_1))| + k^-(Q_2 \cup Q_3)$. Hence, it remains to upper bound $|Q_4|$ with $Q_4 = \mathrm{ALG} \setminus (Q_1 \cup Q_2 \cup Q_3)$ by $4 \cdot k_h$. By definition, Q_4 only contains edges that are queried in Line 12. Thus, at least half the queries of Q_4 are elements of W that are part of the matching h. By Observation 14, no element of W is part of the partial matching $\bar{h}$ in Line 10. Hence, in each execution of Line 12, at least half the queries are not part of $\bar{h}$ in Line 10 but added to h in Line 11. Our goal is to bound the number of such elements.

We start with some definitions. Define G_j as the problem instance at the j'th execution of Line 11, and let G_0 denote the initial problem instance. For each G_j, let $\bar{G}_j = (\bar{V}_j, \bar{E}_j)$, T_L^j and T_U^j denote the corresponding vertex cover instance and lower and upper limit trees. Observe that each G_j has unique $T_L^j = T_U^j$, and, by Lemma 5, is prediction mandatory free. Let Y_j denote the set of queries made by the algorithm to transform instance G_{j-1} into instance G_j. We partition Q_4 into subsets S_j, where S_j contains the edges of Q_4 that are queried in the j'th execution of Line 12. We claim $|S_j| \leq 4 \cdot k^+(Y_j)$ for each j, which implies $|Q_4| \leq \sum_j |S_j| \leq 4 \cdot \sum_j k^+(Y_j) \leq 4 \cdot k_h$. To show the claim, we rely on the following lemma.

▶ **Lemma 17.** *Let l, f be non-trivial edges in G_j such that $\{l, f\} \in \bar{E}_{j-1} \Delta \bar{E}_j$, then $k^-(l), k^-(f) \geq 1$. Furthermore, $k^+(Y_j) \geq |U|$ for the set U of all endpoints of such edges $\{l, f\}$.*

▶ **Lemma 18.** $|\mathrm{ALG}| \leq |\mathrm{OPT}| + 5 \cdot k_h$.

Proof. We show $|S_j| \leq 4 \cdot k^+(Y_j)$ for each j, which, in combination with Lemma 15, implies the lemma. Consider an arbitrary S_j and the corresponding set Y_j. Further, let h_{j-1} denote the maximum matching computed and used by the algorithm for vertex cover instance $\bar{G}_{j-1}$, and let $\bar{h}_{j-1} = \{\{e, e'\} \in h_{j-1} \mid \{e, e'\} \in \bar{E}_j\}$. Finally, let h_j denote the matching that the algorithm uses for vertex cover instance $\bar{G}_j$. That is, h_j is computed by completing $\bar{h}_{j-1}$ with a standard augmenting path algorithm. As already argued, at least half the elements of S_j are not matched by $\bar{h}_{j-1}$ but are matched by h_j (cf. Observation 14).

We bound the number of such elements by exploiting that h_j is constructed from $\bar{h}_{j-1}$ via a standard augmenting path algorithm. By definition, each iteration of the augmenting path algorithm increases the size of the current matching (starting with $\bar{h}_{j-1}$) by one and, in doing so, matches two new elements. In total, at most $2 \cdot (|h_j| - |\bar{h}_{j-1}|)$ previously unmatched elements become part of the matching. Thus, $|S_j| \leq 4 \cdot (|h_j| - |\bar{h}_{j-1}|)$.

We show $(|h_j| - |\bar{h}_{j-1}|) \leq k^+(Y_j)$. According to *Kőnig-Egerváry*'s Theorem (e.g., [14]), the size of h_j is equal to the size $|VC_j|$ of the minimum vertex cover for $\bar{G}_j$. We show $|VC_j| \leq |\bar{h}_{j-1}| + k^+(Y_j)$, which implies $(|h_j| - |\bar{h}_{j-1}|) = |VC_j| - |\bar{h}_{j-1}| \leq k^+(Y_j)$, and, thus, the claim. Let $\overline{VC}_{j-1} = \{e \in VC_{j-1} \mid \exists e' \text{ s.t. } \{e, e'\} \in \bar{h}_{j-1}\}$, then $|\overline{VC}_{j-1}| = |\bar{h}_{j-1}|$.

We prove that we can construct a vertex cover for $\bar{G}_j$ by adding at most $k^+(Y_j)$ elements to $\overline{VC}_{j-1}$, which implies $|VC_j| \leq |\bar{h}_{j-1}| + k^+(Y_j)$. Consider vertex cover instance $\bar{G}_j$ and set $\overline{VC}_{j-1}$. By definition, $\overline{VC}_{j-1}$ covers all edges that are part of partial matching $\bar{h}_{j-1}$.

Consider the elements of $\bar{V}_j$ that are an endpoint of an edge in $\{e,f\} \in \bar{E}_j \Delta \bar{E}_{j-1}$ with e,f non-trivial in G_j. By Lemma 17, $k^-(e) \geq 1$ for each such e and $k^+(Y_j) \geq |U|$ for the set U of all such elements. Thus, we can afford to add U to the vertex cover.

Next, consider an edge $\{e,f\} \in \bar{E}_j$ that is not covered by $\overline{VC}_{j-1} \cup U$. Since $\{e,f\}$ is not covered by U, it must hold that $\{e,f\} \in \bar{E}_j \cap \bar{E}_{j-1}$. Thus, $\{e,f\}$ was covered by VC_{j-1} but is not covered by $\overline{VC}_{j-1}$. This implies $\{e,f\} \cap VC_{j-1} \neq \emptyset$ but $\{e,f\} \cap \overline{VC}_{j-1} = \emptyset$. Assume w.l.o.g. that $e \in VC_{j-1}$. Then, there must be an e' such that $\{e,e'\} \in h_{j-1}$ but $\{e,e'\} \notin \bar{h}_{j-1}$. It follows that $\{e,e'\} \notin \bar{E}_j$. As $\{e,f\}$ is not covered by U, the endpoint e' must be trivial in G_j but non-trivial in G_{j-1}. Thus, e' must have been queried (i) as a mandatory element (or a matching partner) in Line 6 or 13, (ii) as part of VC_{j-1} in Line 5 or (iii) in Line 12. Case (ii) implies $e' \in VC_{j-1}$, contradicting $e \in VC_{j-1}$. Cases (i) or (iii) imply a query to the matching partner e of e', which contradicts $\{e,f\} \in \bar{E}_j$ (as e would be trivial). Thus, $\{e,f\}$ is covered by $\overline{VC}_{j-1} \cup U$, which implies that $\overline{VC}_{j-1} \cup U$ is a vertex cover for $\bar{G}_j$. Lemma 17 implies $|U| \leq k^+(Y_j)$. So, $|VC_j| \leq |\bar{h}_{j-1}| + k^+(Y_j)$ which concludes the proof. ◀

Lemmas 16 and 18 imply Theorem 13. Combining Theorems 6 and 13, we show Theorem 12.

6 Further research directions

Plenty other (optimization) problems seem natural in the context of explorable uncertainty with untrusted predictions. For our problem, it would be nice to close the gap in the robustness. We expect that our results extend to all matroids as it does in the classical setting. While we ask for the minimum number of queries to solve a problem *exactly*, it is natural to ask for approximate solutions. The bad news is that for the MST problem there is no improvement over the robustness guarantee of 2 possible even when allowing an arbitrarily large approximation of the exact solution [43, Section 10]. However, it remains open whether an improved consistency or an error-dependent competitive ratio are possible.

References

1 Mohamed Abdel-Nasser, Karar Mahmoud, Osama A. Omer, Matti Lehtonen, and Domenec Puig. Link quality prediction in wireless community networks using deep recurrent neural networks. *Alexandria Engineering Journal*, 59(5):3531–3543, 2020. doi:10.1016/j.aej.2020.05.037.

2 Susanne Albers and Alexander Eckl. Explorable uncertainty in scheduling with non-uniform testing times. In *WAOA*, volume 12806 of *Lecture Notes in Computer Science*, pages 127–142. Springer, 2020. doi:10.1007/978-3-030-80879-2_9.

3 Susanne Albers and Alexander Eckl. Scheduling with testing on multiple identical parallel machines. In *WADS*, volume 12808 of *Lecture Notes in Computer Science*, pages 29–42. Springer, 2021. doi:10.1007/978-3-030-83508-8_3.

4 Spyros Angelopoulos, Christoph Dürr, Shendan Jin, Shahin Kamali, and Marc P. Renault. Online computation with untrusted advice. In *ITCS*, volume 151 of *LIPIcs*, pages 52:1–52:15, 2020. doi:10.4230/LIPIcs.ITCS.2020.52.

5 Antonios Antoniadis, Christian Coester, Marek Eliás, Adam Polak, and Bertrand Simon. Online metric algorithms with untrusted predictions. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 345–355. PMLR, 2020. URL: http://proceedings.mlr.press/v119/antoniadis20a.html.

6 Antonios Antoniadis, Themis Gouleakis, Pieter Kleer, and Pavel Kolev. Secretary and online matching problems with machine learned advice. In *NeurIPS*, 2020.

7 Luciana Arantes, Evripidis Bampis, Alexander V. Kononov, Manthos Letsios, Giorgio Lucarelli, and Pierre Sens. Scheduling under uncertainty: A query-based approach. In *IJCAI*, pages 4646–4652. ijcai.org, 2018. doi:10.24963/ijcai.2018/646.

8 Sepehr Assadi, Deeparnab Chakrabarty, and Sanjeev Khanna. Graph connectivity and single element recovery via linear and OR queries. In *ESA*, volume 204 of *LIPIcs*, pages 7:1–7:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ESA.2021.7.

9 Yossi Azar, Stefano Leonardi, and Noam Touitou. Flow time scheduling with uncertain processing time. In *STOC*, pages 1070–1080. ACM, 2021. doi:10.1145/3406325.3451023.

10 Evripidis Bampis, Konstantinos Dogeas, Alexander V. Kononov, Giorgio Lucarelli, and Fanny Pascual. Speed scaling with explorable uncertainty. In *SPAA*, pages 83–93. ACM, 2021. doi:10.1145/3409964.3461812.

11 Evripidis Bampis, Christoph Dürr, Thomas Erlebach, Murilo Santos de Lima, Nicole Megow, and Jens Schlöter. Orienting (hyper)graphs under explorable stochastic uncertainty. In *ESA*, volume 204 of *LIPIcs*, pages 10:1–10:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ESA.2021.10.

12 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge estimation with independent set oracles. *ACM Trans. Algorithms*, 16(4):52:1–52:27, 2020.

13 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge estimation with independent set oracles. *ACM Trans. Algorithms*, 16(4):52:1–52:27, 2020. doi:10.1145/3404867.

14 Norman Biggs, E Keith Lloyd, and Robin J Wilson. *Graph Theory, 1736-1936*. Oxford University Press, 1986.

15 Richard Bruce, Michael Hoffmann, Danny Krizanc, and Rajeev Raman. Efficient update strategies for geometric computing with uncertainty. *Theory Comput. Syst.*, 38(4):411–423, 2005. doi:10.1007/s00224-004-1180-4.

16 Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012. doi:10.1561/2200000024.

17 Steven Chaplick, Magnús M. Halldórsson, Murilo S. de Lima, and Tigran Tonoyan. Query minimization under stochastic uncertainty. *Theor. Comput. Sci.*, 895:75–95, 2021. doi:10.1016/j.tcs.2021.09.032.

18 Christoph Dürr, Thomas Erlebach, Nicole Megow, and Julie Meißner. An adversarial model for scheduling with testing. *Algorithmica*, 82(12):3630–3675, 2020. doi:10.1007/s00453-020-00742-2.

19 Franziska Eberle, Alexander Lindermayr, Nicole Megow, Lukas Nölke, and Jens Schlöter. Robustification of online graph exploration methods. In *AAAI*, 2022.

20 Thomas Erlebach, Murilo Santos de Lima, Nicole Megow, and Jens Schlöter. Learning-augmented query policies for minimum spanning tree with uncertainty. *CoRR*, abs/2206.15201, 2022.

21 Thomas Erlebach and Michael Hoffmann. Minimum spanning tree verification under uncertainty. In *WG 2014: International Workshop on Graph-Theoretic Concepts in Computer Science*, volume 8747 of *Lecture Notes in Computer Science*, pages 164–175. Springer, 2014. doi:10.1007/978-3-319-12340-0_14.

22 Thomas Erlebach and Michael Hoffmann. Query-competitive algorithms for computing with uncertainty. *Bull. EATCS*, 116, 2015. doi:10.1016/j.tcs.2015.11.025.

23 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. Query-competitive algorithms for cheapest set problems under uncertainty. *Theor. Comput. Sci.*, 613:51–64, 2016.

24 Thomas Erlebach, Michael Hoffmann, Danny Krizanc, Matús Mihalák, and Rajeev Raman. Computing minimum spanning trees with uncertainty. In *STACS*, volume 1 of *LIPIcs*, pages 277–288. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany, 2008. doi:10.4230/LIPIcs.STACS.2008.1358.

25 Tomás Feder, Rajeev Motwani, Liadan O'Callaghan, Chris Olston, and Rina Panigrahy. Computing shortest paths with uncertainty. *J. Algorithms*, 62(1):1–18, 2007. doi:10.1016/j.jalgor.2004.07.005.

26 Tomás Feder, Rajeev Motwani, Rina Panigrahy, Chris Olston, and Jennifer Widom. Computing the median with uncertainty. *SIAM J. Comput.*, 32(2):538–547, 2003. doi:10.1137/S0097539701395668.

27 Jacob Focke, Nicole Megow, and Julie Meißner. Minimum spanning tree under explorable uncertainty in theory and experiments. *ACM J. Exp. Algorithmics*, 25:1–20, 2020. doi:10.1145/3422371.

28 John Gittins, Kevin Glazebrook, and Richard Weber. *Multi-armed Bandit Allocation Indices*. Wiley, 2nd edition, 2011.

29 Marc Goerigk, Manoj Gupta, Jonas Ide, Anita Schöbel, and Sandeep Sen. The robust knapsack problem with queries. *Comput. Oper. Res.*, 55:12–22, 2015. doi:10.1016/j.cor.2014.09.010.

30 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.

31 Sreenivas Gollapudi and Debmalya Panigrahi. Online algorithms for rent-or-buy with expert advice. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 2319–2327. PMLR, 2019.

32 Anupam Gupta, Haotian Jiang, Ziv Scully, and Sahil Singla. The markovian price of information. In *IPCO*, volume 11480 of *Lecture Notes in Computer Science*, pages 233–246. Springer, 2019. doi:10.1007/978-3-030-17953-3_18.

33 Manoj Gupta, Yogish Sabharwal, and Sandeep Sen. The update complexity of selection and related problems. *Theory Comput. Syst.*, 59(1):112–132, 2016. doi:10.1007/s00224-015-9664-y.

34 Magnús M. Halldórsson and Murilo Santos de Lima. Query-competitive sorting with uncertainty. *Theor. Comput. Sci.*, 867:50–67, 2022. doi:10.1016/j.tcs.2021.03.021.

35 Simon Kahan. A model for data in motion. In *STOC*, pages 267–277. ACM, 1991. doi:10.1145/103418.103449.

36 Sanjeev Khanna and Wang Chiew Tan. On computing functions with uncertainty. In *PODS*, pages 171–182. ACM, 2001. doi:10.1145/375551.375577.

37 Ravi Kumar, Manish Purohit, Aaron Schild, Zoya Svitkina, and Erik Vee. Semi-online bipartite matching. In *ITCS*, volume 124 of *LIPIcs*, pages 50:1–50:20. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2019. doi:10.4230/LIPIcs.ITCS.2019.50.

38 Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Online scheduling via learned weights. In *SODA*, pages 1859–1877. SIAM, 2020. doi:10.1137/1.9781611975994.114.

39 Alexander Lindermayr, Nicole Megow, and Bertrand Simon. Double Coverage with Machine-Learned Advice. In *ITCS*, volume 215 of *LIPIcs*, pages 99:1–99:18, 2022. doi:10.4230/LIPIcs.ITCS.2022.99.

40 Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. *J. ACM*, 68(4):24:1–24:25, 2021. doi:10.1145/3447579.

41 Takanori Maehara and Yutaro Yamaguchi. Stochastic packing integer programs with few queries. *Math. Program.*, 182(1):141–174, 2020. doi:10.1007/s10107-019-01388-x.

42 Hanna Mazzawi. Optimally reconstructing weighted graphs using queries. In *SODA*, pages 608–615. SIAM, 2010. doi:10.1137/1.9781611973075.51.

43 Nicole Megow, Julie Meißner, and Martin Skutella. Randomization helps computing a minimum spanning tree under uncertainty. *SIAM J. Comput.*, 46(4):1217–1240, 2017. doi:10.1137/16M1088375.

44 Arturo I. Merino and José A. Soto. The minimum cost query problem on matroids with uncertainty areas. In *ICALP*, volume 132 of *LIPIcs*, pages 83:1–83:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.83.

45 Michael Mitzenmacher. Scheduling with predictions and the price of misprediction. In *ITCS*, volume 151 of *LIPIcs*, pages 14:1–14:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.14.

46 Noam Nisan. The demand query model for bipartite matching. In *SODA*, pages 592–599. SIAM, 2021. doi:10.1137/1.9781611976465.36.

47 Chris Olston and Jennifer Widom. Offering a precision-performance tradeoff for aggregation queries over replicated data. In *VLDB*, pages 144–155. Morgan Kaufmann, 2000.

48 Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ML predictions. In *NeurIPS*, pages 9684–9693, 2018.

49 Dhruv Rohatgi. Near-optimal bounds for online caching with machine learned advice. In *SODA*, pages 1834–1845. SIAM, 2020. doi:10.1137/1.9781611975994.112.

50 Aviad Rubinstein, Tselil Schramm, and S. Matthew Weinberg. Computing exact minimum cuts without knowing the graph. In *ITCS*, volume 94 of *LIPIcs*, pages 39:1–39:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ITCS.2018.39.

51 Sahil Singla. The price of information in combinatorial optimization. In *SODA*, pages 2523–2532. SIAM, 2018. doi:10.1137/1.9781611975031.161.

52 William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. doi:10.2307/2332286.

53 Alexander Wei. Better and simpler learning-augmented online caching. In *APPROX-RANDOM*, volume 176 of *LIPIcs*, pages 60:1–60:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.60.

54 Alexander Wei and Fred Zhang. Optimal robustness-consistency trade-offs for learning-augmented online algorithms. In *NeurIPS*, 2020.

55 Martin Weitzman. Optimal search for the best alternative. *Econometrica*, 47(3):641–54, 1979. doi:10.2307/1910412.