



# Efficient Algorithms for Minimax Decisions Under Tree-Structured Incompleteness

Thijs van Ommen<sup>1(✉)</sup>, Wouter M. Koolen<sup>2</sup>, and Peter D. Grünwald<sup>2,3</sup>

<sup>1</sup> Utrecht University, Utrecht, The Netherlands  
t.vanommen@uu.nl

<sup>2</sup> Centrum Wiskunde & Informatica, Amsterdam, The Netherlands  
{wmkoolen, pdg}@cwi.nl

<sup>3</sup> Leiden University, Leiden, The Netherlands

**Abstract.** When decisions must be based on incomplete (coarsened) observations and the coarsening mechanism is unknown, a minimax approach offers the best guarantees on the decision maker's expected loss. Recent work has derived mathematical conditions characterizing minimax optimal decisions, but also found that computing such decisions is a difficult problem in general. This problem is equivalent to that of maximizing a certain conditional entropy expression. In this work, we present a highly efficient algorithm for the case where the coarsening mechanism can be represented by a tree, whose vertices are outcomes and whose edges are coarse observations.

**Keywords:** Coarse data · Incomplete observations ·  
Minimax decision making · Maximum entropy

## 1 Introduction

Suppose a decision maker needs to choose an action  $a$ , and will suffer an amount of loss determined by  $a$  and an unobserved random variable  $X$ . The decision maker knows the distribution of  $X$ , and receives some information on the realized value  $X = x$  in the form of a coarsened observation: a set  $Y = y$  that includes  $x$  but also other, unrealized outcomes. Here,  $x$  lies in a finite set  $\mathcal{X}$ , and  $y$  is a member of some family  $\mathcal{Y} \subset 2^{\mathcal{X}}$ ; both  $\mathcal{X}$  and  $\mathcal{Y}$  are also known to the decision maker, but importantly, the distribution  $P(Y | X)$  of the coarsening mechanism is not.

One of the most well-known examples illustrating this setting is the Monty Hall puzzle [14]: In a game show, the contestant is faced with three doors  $\mathcal{X} = \{1, 2, 3\}$ .  $X$  indicates which of these hides a prize. The contestant initially picks a door; we will assume w.l.o.g. this is door 2. Then the quizmaster opens either door 1 or 3, revealing a goat. When both doors could be opened (i.e. if  $X = 2$ ), one is chosen by the quizmaster's unknown coarsening mechanism. (In our

notation,  $Y$  is the set of the two doors that are still closed at this point; its possible values are the two members of  $\mathcal{Y} = \{\{1, 2\}, \{2, 3\}\}$ .) The contestant is now offered the option to switch to another door. The surprising insight is that the strategy of switching doors results in a larger probability of winning the prize.

We adopt a minimax (or worst-case) approach: we want to find a strategy (a function that maps each  $y \in \mathcal{Y}$  to an action) for which the maximum expected loss over all possible coarsening mechanisms is as small as possible. Such a strategy does not require us to make any assumptions on the coarsening mechanism, so is a robust choice when the mechanism is unknown. (In the case of the Monty Hall puzzle, this means we do not need to assume anything about the distribution of  $Y$  given  $X = 2$ .)

In this paper, we propose efficient algorithms for a decision problem that generalizes the Monty Hall puzzle in the following way: to any number of outcomes, any distribution  $P(X)$  over them, a very general class of loss functions, and any family  $\mathcal{Y}$  that is the set of edges of an undirected tree over vertices  $\mathcal{X}$ . In other words, each set  $y \in \mathcal{Y}$  consists of two elements of  $\mathcal{X}$ , and for each pair  $x_a, x_b \in \mathcal{X}$ , there exists a unique sequence  $(x_i)_{i=1}^k$  of distinct elements from  $\mathcal{X}$  with  $x_1 = x_a$ ,  $x_k = x_b$ , and  $\{x_1, x_2\}, \{x_2, x_3\}, \dots, \{x_{k-1}, x_k\} \in \mathcal{Y}$ . We will call  $\mathcal{Y}$  the *message structure*, and its elements  $y \in \mathcal{Y}$  *messages*.

To illustrate this generalization, consider a version of the Monty Hall game show with a row of doors  $\mathcal{X} = \{1, 2, \dots, n\}$ , where the quizmaster will pick two adjacent doors that he leaves shut, revealing a goat behind each other door (so  $\mathcal{Y} = \{\{i, i+1\} \mid i = 1, \dots, n-1\}$ ). If the number of doors  $n$  is odd and the distribution on  $X$  is uniform, we find (see Sect. 2) that, upon observing  $Y = y$ , a cautious decision maker should assign probability  $(n+1)/(2n) > 1/2$  to the door in  $y$  with the odd index, and  $(n-1)/(2n) < 1/2$  to the one with the even index. The case  $n = 3$  is the original Monty Hall puzzle, where a contestant who always switches to door 1 or 3 will have a probability of  $2/3$  of winning the prize. The generalization we consider also extends to distributions over  $X$  other than uniform, for which the problem becomes computationally trickier.

The above message structure  $\mathcal{Y}$  (which we call a *path graph*) may occur in practice as the message structure of a decision problem when, for example, a real-valued quantity of interest is reported to us as an integer, but we do not know if the value was rounded up or down. Then outcomes  $x \in \mathcal{X}$  correspond to the intervals  $(a_i, a_i + 1)$  between consecutive integers (we assume that the true value is a.s. not an integer), and messages to unions of two adjacent intervals.

This type of decision problem with incomplete information was introduced in [16], where minimax optimal strategies are characterized for arbitrary  $\mathcal{Y}$ , but where the question of how to compute them is not addressed. This computational problem was considered previously in [17], where it was demonstrated that the problem is hard for general  $\mathcal{Y}$ , but a direct formula could be given for finding a minimax optimal strategy in the special case that  $\mathcal{Y}$  forms a partition matroid.<sup>1</sup>

<sup>1</sup> We refer to that paper for a definition, but remark that if each message consists of two outcomes, the class of partition matroids coincides with complete bipartite graphs. The message structure of the Monty Hall puzzle is an example.

Related work dealing with coarse data sometimes proceeds by making assumptions about the coarsening mechanism; for example, the CAR (coarsened at random) assumption [9] or the superset assumption [10] (see Sect. 2 for an explanation). Neither of these assumptions is compatible with the Monty Hall setting; see e.g. [5, 7].

The approach we consider is more closely related to the maximin strategy described in [3] and studied specifically in [8]. However, it differs in some respects, e.g.: our objective does not feature a marginal or joint, but a conditional distribution of outcomes given messages (see (1)); and our objective is not interpreted as a (generalized) likelihood function, as we do not have a data set. This reflects that we are interested in making a decision pertaining to an unknown outcome *given a single message* (which is a coarsened observation of the outcome).

The general problem of finding a minimax optimal strategy can be solved using convex optimization. Reasonably efficient algorithms exist for this task [2], but they converge to the solution rather than computing it exactly. The algorithms in this paper are *strongly polynomial* [13]. A strongly polynomial algorithm finds the exact solution in a number of steps polynomial in the number of elements in the input, regardless of the precision of any numeric elements.

The rest of this paper is structured as follows. In Sect. 2, relevant results from [16] are summarized. We consider in Sect. 3 the special case where  $(\mathcal{X}, \mathcal{Y})$  is a path graph, where the solution is given by a surprising and intuitive algorithm. This algorithm is extended to the case of arbitrary trees in Sect. 4. All proofs are in the appendix. The results on path graphs were previously described in [15]; the results on trees have not appeared elsewhere.

## 2 Preliminaries

A *decision problem with tree-structured incompleteness* is given by a finite set  $\mathcal{X}$ , a family  $\mathcal{Y}$  of two-element subsets of  $\mathcal{X}$  such that the undirected graph  $(\mathcal{X}, \mathcal{Y})$  forms a tree, a distribution  $p$  over  $\mathcal{X}$  having  $p_x > 0$  everywhere, and a loss function  $L : \mathcal{X} \times \mathcal{A} \rightarrow [0, \infty]$ , where  $\mathcal{A}$  is the set of actions available to the decision maker. We assume that the loss function  $L$  satisfies the conditions in Theorem 18 of [16].<sup>2</sup> A *coarsening mechanism* for this problem is an (unknown) joint distribution  $P$  on  $\mathcal{X} \times \mathcal{Y}$  that satisfies  $P(x, y) = 0$  whenever  $x \notin y$ , and  $P(x) = \sum_{y \ni x} P(x, y) = p_x$  for each  $x$ .

The minimax approach may be viewed as a game: first the decision maker chooses a strategy  $A : \mathcal{Y} \rightarrow \mathcal{A}$ , then the opponent chooses a coarsening mechanism  $P$ , and finally the decision maker's expected loss  $\sum_{x,y} P(x, y)L(x, A(y))$  is evaluated. The opponent's goal is to make the expected loss as large as possible (i.e. it is a zero-sum game). If the opponent were to move first, their best strategy would be the *maximin optimal* coarsening mechanism. In the case that  $L$  is

<sup>2</sup> Namely, that the generalized entropy  $H_L$  [6] is finite and continuous; further,  $H_L$  or an affine transformation of it is invariant under permutation of  $x_1$  and  $x_3$  whenever  $\{x_1, x_2\}, \{x_2, x_3\} \in \mathcal{Y}$ . See [16] for definitions.

logarithmic loss, this is the  $P$  that maximizes the expected conditional entropy

$$\sum_{y \in \mathcal{Y}} P(y) H(P(\cdot | y)) = \sum_{y \in \mathcal{Y}} P(y) \sum_{x \in \mathcal{Y}} -P(x | y) \log P(x | y). \tag{1}$$

It was found in [16] that if the action space is rich enough, this game has a Nash equilibrium, so that neither player benefits from knowing the other’s strategy before picking their own. We concentrate on finding a maximin  $P$ , because once it is known, minimax optimal strategies  $A$  are typically easily determined.

For the setting considered in this paper, it was shown in [16, Theorem 18] that a coarsening mechanism  $P$  is maximin optimal if for some vector  $q \in [0, 1]^{\mathcal{X}}$  it satisfies the *RCAR condition*:

$$\begin{aligned} q_x &= P(x | y) \text{ for all } y \in \mathcal{Y}, x \in y \text{ with } P(y) > 0, \text{ and} \\ \sum_{x \in y} q_x &\leq 1 \text{ for all } y \in \mathcal{Y}. \end{aligned} \tag{2}$$

Note that the second equation holds with equality for  $y$  with  $P(y) > 0$ . The vector  $q$  is called the RCAR vector; it exists and is unique by [16, Lemma 11]. We remark that the loss function  $L$  does not feature in this condition; this implies that a coarsening mechanism  $P$  that satisfies the RCAR condition is maximin optimal *regardless* of what loss function we are interested in,<sup>3</sup> and that the entropy maximization (1) is relevant for any  $L$ , not just logarithmic loss.

RCAR stands for ‘reverse CAR’, because the first line of (2) mimics the form of the CAR assumption [9], but with  $x$  and  $y$  switched. It is also similar in form to (part of) the superset assumption [10], but both CAR and the superset assumption look at  $P(y | x)$ , while RCAR considers  $P(x | y)$ . But the most crucial difference with these is that RCAR is not an assumption, but rather a condition we can check to verify if a coarsening mechanism is maximin optimal.

The question is: how do we find maximin optimal/RCAR coarsening mechanism? One computational challenge in finding the maximin optimal coarsening mechanism is that for some coarsening mechanisms  $P$  (including the maximin one), some  $y$  may have  $P(y) = 0$ . At such a point  $P$ , that  $y$ ’s contribution to the expected conditional entropy is nondifferentiable. This means for instance that standard convex optimization algorithms will converge slowly to such a point. The algorithms in this paper overcome this challenge.

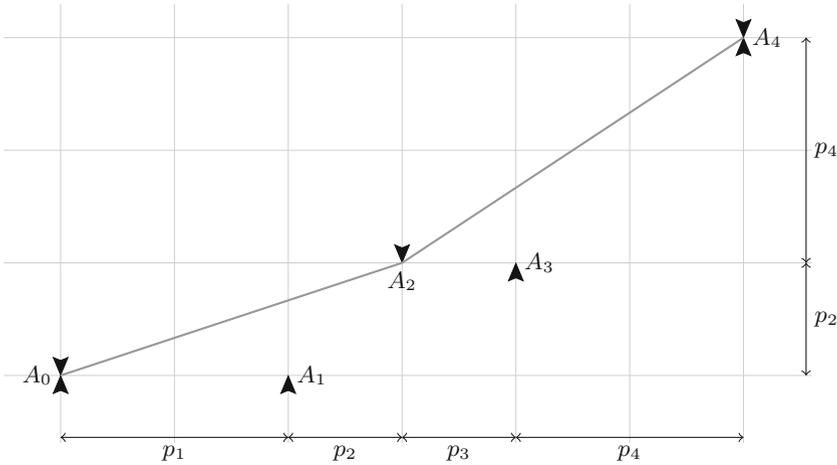
### 3 Path Graphs and the Taut String Algorithm

Even though the results in this section will be superseded later when we give an algorithm for general trees, we devote this section to the subclass of decision problems for which the graph  $(\mathcal{X}, \mathcal{Y})$  consists of just a single path. The reason is that for such problems, the minimax optimal coarsening mechanisms turn out to be described by an intuitive physical problem, for which an efficient algorithm is already known.

<sup>3</sup> This property is called *loss invariance*: see [16, Section 5.5].

### 3.1 Correspondence

Consider a decision problem where the messages form a path: for the  $n \geq 2$  outcomes  $\mathcal{X} := \{1, 2, \dots, n\}$ , the messages  $\mathcal{Y}$  are  $y_1 = \{1, 2\}, \dots, y_{n-1} = \{n-1, n\}$ . (A graph of the form  $(\mathcal{X}, \mathcal{Y})$  is called a *path graph*.) Then the solution corresponds to that of a *taut string problem*. Imagine a string is constrained to pass above certain points (say, pins on a board), and below others. Then the string is pulled taut. The taut string will follow the shortest allowed path between its endpoints, going in straight line segments between the points it is pushed against.



**Fig. 1.** The taut string problem corresponding to the decision problem with  $\mathcal{X} = \{1, 2, 3, 4\}$ ,  $\mathcal{Y} = \{y_1, y_2, y_3\}$  with  $y_i = \{i, i + 1\}$ , and marginal on the outcomes  $p = (1/3, 1/6, 1/6, 1/3)$ . The arrowheads at the points  $A_0, \dots, A_4$  show on what side the string must pass. We see that the string, when pulled taut, touches the point  $A_2$ ; its slope is  $1/3$  to the left of  $A_2$  and  $2/3$  to the right.

Taut strings have been considered in the statistics literature before; see for example [1, 4, 11], where taut strings appear as a way of defining simple functions approximating noisy regression data. In these applications, all pins come in pairs, but we do not restrict the placement of pins in this way.

The taut string problem we are interested in uses the constraining points  $A_0, A_1, \dots, A_n$ , with  $A_0 = (0, 0)$  and

$$A_k = \left( \sum_{i \leq k} p_i, \sum_{\substack{i \leq k, \\ i \text{ even}}} p_i \right) \tag{3}$$

for  $k \in \{1, \dots, n\}$ . The string must pass through the points  $A_0$  and  $A_n$ ; above points  $A_k$  with  $k$  odd; and below  $A_k$  for  $k$  even. See Fig. 1 for an example.

The following theorem relates the solution of this taut string problem instance to the maximin optimal  $P$  (the proof is in the appendix).

**Theorem 1.** *Given a decision problem on a path graph, find the solution of the taut string problem described in (3). Then a maximin optimal coarsening mechanism  $P$  is given by:*

- For  $0 < k < n$  such that the string touches the point  $A_k$ , we have  $P(y_k) = 0$ ;
- For  $0 < k < n$  such that the string does not touch  $A_k$ ,  $P(y_k) = |\delta_k| / (\alpha_k(1 - \alpha_k))$ , where  $\delta_k$  is the vertical distance between  $A_k$  and the string, and  $\alpha_k$  is the slope of the string at that point;
- Also for  $k$  such that the string does not touch the point  $A_k$  (so  $P(y_k) > 0$ ), the conditional distribution  $P(\cdot | y_k)$  puts mass on the even outcome in  $y_k$  equal to the slope of the string as it passes above or below  $A_k$ .

For the decision problem and corresponding taut string problem displayed in Fig. 1, we conclude that:

- $P(y_2) = 0$ ;
- For  $y_1$ ,  $|\delta_1| = 1/9$  (it is two-thirds of  $p_2 = 1/6$ ) and  $\alpha_1 = 1/3$ , so  $P(y_1) = (1/9)/(9/2) = 1/2$ . In the same way, we find  $P(y_3) = 1/2$ .
- Using that the slope of the string above  $A_1$  equals  $1/3$ , we find  $P(2 | y_1) = 1/3$  (so  $P(1 | y_1) = 2/3$ ). Above  $A_3$ , the slope equals  $2/3$ , so  $P(4 | y_3) = 2/3$  and  $P(3 | y_3) = 1/3$ .

### 3.2 Algorithm

We can now find maximin optimal coarsening mechanisms for decision problems on path graphs efficiently using the taut string algorithm, in  $O(n)$  time [4]. This is clearly much more efficient than using a general purpose convex optimization algorithm. We list the taut string algorithm in Algorithm 1; for a more detailed explanation, we refer to [4] and [1].

The algorithm keeps track of three sequences of points. These represent piecewise linear functions:  $\mathbf{K}$  is the solution, specified as the sequence of points the taut string pushes against;  $\mathbf{G}$  is the *greatest convex minorant* (the pointwise maximum convex function respecting the upper bounds) of the part of the input that has been read but not added to the solution yet: and  $\mathbf{S}$  the *smallest concave majorant* of that part of the input. Each of these sequences is a subsequence of the input points  $A_0, \dots, A_n$ , and at each step of the algorithm, the first points of  $\mathbf{G}$  and  $\mathbf{S}$  are equal to the last point of  $\mathbf{K}$ . The algorithm operates only on the beginning and end of each of these sequences, so these operations can be implemented efficiently without the aid of complex data structures.

We denote the number of elements in a sequence by  $|\mathbf{K}|$ , use zero-based indices (so  $\mathbf{K}[0]$  is the first element of  $\mathbf{K}$ ), and use negative indices to refer to the end of a sequence:  $\mathbf{K}[-1]$  is the last element,  $\mathbf{K}[-2]$  the second-to-last, etc. We write  $\alpha(A_i, A_j)$  for the slope of the line segment from  $A_i$  to  $A_j$ , with  $i < j$ . For the points (3) used in the taut string problem corresponding to a decision problem, these slopes are given by  $\alpha(A_i, A_j) = \sum_{i < k \leq j, k \text{ even}} p_k / \sum_{i < k \leq j} p_k$ .

---

**Algorithm 1:** The taut string algorithm [4]

---

**Input:** Points  $A_0, \dots, A_n$  constraining the string, sorted from left to right**Output:** Sequence of points  $K$  where the taut string pushes against the constraintsLet the sequences  $G$ ,  $S$ , and  $K$  each consist of the single point  $A_0$ ;**for**  $i$  from 1 to  $n$  **do**    **if**  $i$  is even or  $i = n$  **then**        //  $A_i$  is an upper bound        Append  $A_i$  to  $G$ ;        **while**  $|G| \geq 3$  and  $\alpha(G[-3], G[-2]) \geq \alpha(G[-2], G[-1])$  **do**            Delete second-to-last point from  $G$ ;        **end**        //  $G$  is now convex    **end**    **if**  $i$  is odd or  $i = n$  **then**        //  $A_i$  is a lower bound        Append  $A_i$  to  $S$ ;        **while**  $|S| \geq 3$  and  $\alpha(S[-3], S[-2]) \leq \alpha(S[-2], S[-1])$  **do**            Delete second-to-last point from  $S$ ;        **end**        //  $S$  is now concave    **end**    **while**  $|G| \geq 2$ ,  $|S| \geq 2$ , and  $\alpha(G[0], G[1]) < \alpha(S[0], S[1])$  **do**        // No straight path remains between  $G$  and  $S$         Remove first point from  $G$  and from  $S$ ;        **if** (new) first point in  $G$  is to the left of first point in  $S$  **then**            Append first point in  $G$  to  $K$ , and prepend it to  $S$ ;        **else**            Append first point in  $S$  to  $K$ , and prepend it to  $G$ ;        **end**    **end****end**Append  $A_n$  to  $K$ .

---

## 4 Generalization to Trees

Our main contribution is the generalization of the taut string algorithm to tree-shaped coarsening mechanisms. The result is displayed as Algorithm 2.

### 4.1 Mathematical Description of the Algorithm

Algorithm 2 takes an arbitrary node  $r \in \mathcal{X}$  to be the root of the tree. In reference to this root, we write  $\text{Ch}(x)$  for the children of node  $x$ ,  $\text{pa}(x)$  for the (unique) parent of  $x \neq r$ , and  $\text{De}(x)$  for the descendants of  $x$ , which include  $x$  itself.

The  $f_x$  and  $\bar{f}_x$  in the algorithm are piecewise linear functions from  $[0, 1]$  to  $[0, 1]$ , which allows them to be stored efficiently; see Sect. 4.2 for details. If the

graph is a path and one of its endpoint is chosen as the root, the computations can be simplified to those of Algorithm 1.

**Theorem 2.** *The  $P$  computed by Algorithm 2 is a coarsening mechanism that satisfies the RCAR condition (2).*

---

**Algorithm 2:** The taut tree algorithm

---

Treating the tree as a bipartite graph, call one of its parts  $S$ ;  
 Pick an arbitrary node  $r \in \mathcal{X}$  to be the root of the tree;  
 Recursively determine:

$$\begin{aligned}
 p_x^S &= \sum_{x' \in S \cap \text{De}(x)} p_{x'}; \\
 f_x(\alpha) &= p_x \alpha + \sum_{c \in \text{Ch}(x)} \bar{f}_c(\alpha); \\
 \bar{f}_x(\alpha) &= \begin{cases} \min(f_x(\alpha), p_x^S) & \text{if } x \in S; \\ \max(f_x(\alpha), p_x^S) & \text{if } x \notin S; \end{cases}
 \end{aligned}$$

Find  $\alpha_r$  such that  $f_r(\alpha_r) = p_r^S$ , and  $\alpha_x$  such that  $f_x(\alpha_x) = \bar{f}_x(\alpha_{\text{pa}(x)})$ ;  
 For  $x \neq r$ , let  $P(\{x, \text{pa}(x)\}) = \begin{cases} \frac{1}{\alpha_x(1-\alpha_x)} |\bar{f}_x(\alpha_x) - p_x^S| & \text{if } \alpha_x = \alpha_{\text{pa}(x)}; \\ 0 & \text{otherwise;} \end{cases}$   
 Let  $P(x | \cdot) = \begin{cases} \alpha_x & \text{for } x \in S; \\ 1 - \alpha_x & \text{for } x \notin S. \end{cases}$

---

**4.2 Efficient Implementation of the Algorithm**

An efficient algorithm would require a data structure that allows us to construct the piecewise linear function  $f_r$  (or enough of it to determine  $\alpha_r$ ) quickly. For computing the  $\alpha_x$ 's afterwards, it suffices to store in each node the value of  $\alpha$  where  $f_x(\alpha) = p_x^S$ .

A function  $f$  can be represented using a double-ended priority queue whose elements represent the bends in  $f$ , keyed by  $\alpha$  and with values equal to the change in slope at that point. Alongside this priority queue,  $f(0)$ ,  $f(1)$ ,  $f'(0)$  and  $f'(1)$  are stored. Because the function is increasing (see the proof of Theorem 2), taking a min (max) with a constant of an increasing function represented this way can be done by testing and discarding the smallest (largest) element repeatedly. Summing two functions requires merging their priority queues, so ideally we would use a priority queue that supports an efficient merge operation. By using for example a pointer-based min-max-pair heap [12], a worst-case time complexity of  $O(n \log n)$  can be achieved.

## 5 Conclusion

In this paper, we showed how to efficiently find a coarsening mechanism that maximizes the conditional entropy (1), for two special cases of the message structure  $\mathcal{Y}$ . In the case where  $(\mathcal{X}, \mathcal{Y})$  is a path graph, the problem can be reduced to a taut string problem, and solved in  $O(n)$ . We then generalized this algorithm to the case that  $(\mathcal{X}, \mathcal{Y})$  is a tree, and showed that this allows the solution to be found in  $O(n \log n)$ .

**Acknowledgments.** This research was supported by Vici grant 639.073.04 and Veni grant 639.021.439 from the Netherlands Organization for Scientific Research (NWO).

## Appendix

*Proof (Theorem 1).* Define  $q \in \mathbf{R}^n$  as follows: for  $x$  even, let  $q_x$  equal the slope of the string between  $A_{x-1}$  and  $A_x$ ; for  $x$  odd, let  $q_x$  equal one minus this slope. For any message  $y_k \in \mathcal{Y}$  and outcome  $x \in y_k$  such that the string does not push against  $A_k$ , we see that  $P(x | y) = q_x$  since both are determined by the slope of the string as it passes above or below  $A_k$ . Thus  $P$  is RCAR with vector  $q$ .

For any message  $y_k$  with  $P(y_k) = 0$ , we need to verify that  $P$  satisfies  $q_k + q_{k+1} \leq 1$ . Note that the string does not touch  $A_1$ , because all other points are above the line through  $A_0$  and  $A_1$ . By the same argument (replacing ‘above’ by ‘below’ if  $n$  is odd) the string does not touch  $A_{n-1}$ . If  $k$  is even, the string may be pushed down at  $A_k$ , so the slope to the left of that point, which equals  $q_k$ , must be smaller than or equal to the slope to the right, which equals  $1 - q_{k+1}$ . If  $k$  is odd, we similarly find  $1 - q_k \geq q_{k+1}$ . In both cases, we conclude  $q_k + q_{k+1} \leq 1$ .

What remains is to show that the marginal of  $P$  on the outcomes given in the theorem agrees with  $p$ . We do this by first deriving from  $p$  a formula for the marginal of  $P$  on the messages.

Consider two points  $A_a, A_b$  with  $a < b$  such that the string touches these points but no points in between (thus the string follows a straight line between points  $A_a$  and  $A_b$ ). Using the notation  $p_S$  for  $\sum_{x \in S} p_x$ , the slope of this segment of the string equals

$$\frac{P_{(a,b),\text{even}}}{P_{(a,b)}}$$

This quantity equals  $q_x$  for any even  $a < x \leq b$ , so we call it  $q_{\text{even}}$ , and define  $q_{\text{odd}} := p_{(a,b),\text{odd}}/p_{(a,b)} = 1 - q_{\text{even}}$ .

For  $a < x \leq b$ , the marginal constraints  $\sum_{y \ni x} P(y)P(x | y) = p_x$  are equivalent to  $\sum_{y \ni x} P(y) = p_x/q_x$ . By defining  $P(y_0)$  and  $P(y_n)$  as 0 (note that there are no such elements in  $\mathcal{Y}$ ), we can write  $\sum_{y \ni x} P(y) = P(y_{x-1}) + P(y_x)$ . For  $a < k \leq b$ , we must have  $P(y_k) = p_k/q_k - P(y_{k-1})$  by the marginal constraint on  $x = k$ . Using  $P(y_a) = 0$  and applying this recursion repeatedly, we find that the following choice of marginal on messages satisfies all marginal constraints for  $a < x \leq b$ :

$$P(y_k) = (-1)^k \left( \frac{P_{(a,k),\text{even}}}{q_{\text{even}}} - \frac{P_{(a,k),\text{odd}}}{q_{\text{odd}}} \right) \quad \text{for } a < k \leq b.$$

(Note that we get  $P(y_b) = 0$  as required.) Meanwhile in string land, the point  $A_k$  is at height  $p_{(0,k],\text{even}}$ , and the string intersects the vertical line through  $A_k$  at height  $p_{(0,a],\text{even}} + p_{(a,k]q_{\text{even}}}$ ; the (signed) difference is

$$\begin{aligned} \delta_k &:= p_{(a,k],\text{even}} - p_{(a,k]q_{\text{even}}} = p_{(a,k],\text{even}} - (p_{(a,k],\text{even}} + p_{(a,k],\text{odd}})q_{\text{even}} \\ &= p_{(a,k],\text{even}}q_{\text{odd}} - p_{(a,k],\text{odd}}q_{\text{even}}. \end{aligned}$$

This is positive at even  $k$  where the string passes below  $A_k$ , and negative at odd  $k$ . Thus the choice of marginal we found above equals the choice given in the theorem:

$$P(y_k) = (-1)^k \frac{\delta_k}{q_{\text{even}}q_{\text{odd}}} = \frac{|\delta_k|}{q_{\text{even}}q_{\text{odd}}}$$

which is positive for all  $a < k < b$ . Because  $P$  also satisfies all marginal constraints, it follows that  $P$  is a probability distribution.

*Proof (Theorem 2).* Write  $q_x = P(x \mid \cdot)$ ; we will show that  $q$  is the RCAR vector to  $P$ . We first prove the following claims by induction:

1. for all  $x$ ,  $f_x$  is a strictly increasing function of  $\alpha$ ;
2. for all  $x \neq r$ ,  $\bar{f}_x$  is a nondecreasing function of  $\alpha$ ;
3. for all  $x$  such that either  $x = r$  or  $\alpha_x \neq \alpha_{\text{pa}(x)}$ ,  $f_x(\alpha_x) = p_x^S = \sum_{x' \in \text{De}(x)} p_{x'} \alpha_{x'}$ .

For the first two claims, both the base case and induction step are straightforward.

For the third claim, first observe that if  $x = r$ ,  $\alpha_x$  is chosen to satisfy  $f_x(\alpha_x) = p_x^S$ . The other case is  $\alpha_x \neq \alpha_{\text{pa}(x)}$ ; this happens only if  $f_x(\alpha_{\text{pa}(x)}) \neq \bar{f}_x(\alpha_{\text{pa}(x)})$ , hence if  $\bar{f}_x(\alpha_{\text{pa}(x)}) = p_x^S$ . Then we also see that  $\alpha_x$  is chosen to satisfy  $f_x(\alpha_x) = p_x^S$ . A base case for the induction occurs when  $x$  is such that all  $x' \in \text{De}(x)$  have the same  $\alpha_{x'}$ . In such a base case, we have for all  $x' \in \text{De}(x)$  that  $\bar{f}_{x'}(\alpha_{x'}) = f_{x'}(\alpha_{x'})$ , so that  $f_x(\alpha_x) = \sum_{x' \in \text{De}(x)} p_{x'} \alpha_{x'}$ . For the induction step, we can use that for  $x' \in \text{De}(x)$  with  $\alpha_{x'} \neq \alpha_{\text{pa}(x')}$ , the claim holds by induction. Let  $T_x$  be the descendants of  $x$  that are not descendants of such an  $x'$ . Then for all  $t \in T_x$ , we again have  $\bar{f}_t(\alpha_t) = f_t(\alpha_t)$ , so that the total contribution to  $f_x(\alpha_x)$  from terms  $p_t \alpha$  with  $t \in T_x$  equals  $\sum_{t \in T_x} p_t \alpha_t$ ; combined with the induction hypothesis, we find  $f_x(\alpha_x) = \sum_{x' \in \text{De}(x)} p_{x'} \alpha_{x'}$ . This completes the proof of claim 3.

For given  $x$ , we get  $\sum_{y \ni x} P(y)q_x = p_x$  (and  $P(\{x, x'\}) = 0$  whenever  $q_x + q_{x'} \neq 1$ ) if the marginal distribution  $P(Y)$  satisfies

$$P(\{x, \text{pa}(x)\}) = \begin{cases} \frac{p_x}{q_x} - \sum_{x' \in \text{Ch}(x)} P(\{x', x\}) & \text{if } q_{\text{pa}(x)} = q_x; \\ 0 & \text{otherwise.} \end{cases}$$

For  $x \in S$ , this equals, and for  $x \notin S$ , this equals the negative of,

$$\begin{aligned} \sum_{x' \in T_x \cap S} \frac{p_{x'}}{q_{x'}} - \sum_{x' \in T_x \setminus S} \frac{p_{x'}}{q_{x'}} &= \frac{1 - \alpha_x}{\alpha_x(1 - \alpha_x)} \sum_{x' \in T_x \cap S} p_{x'} - \frac{\alpha_x}{\alpha_x(1 - \alpha_x)} \sum_{x' \in T_x \setminus S} p_{x'} \\ &= \frac{1}{\alpha_x(1 - \alpha_x)} \left[ \sum_{x' \in T_x \cap S} p_{x'} - \alpha_x \sum_{x' \in T_x} p_{x'} \right] = \frac{1}{\alpha_x(1 - \alpha_x)} [p_x^S - f_x(\alpha_x)], \end{aligned}$$

where the final equality follows by applying claim 3 to  $x' \notin T_x$ . We see that this is equal (in both magnitude and sign) to the value assigned to  $P(\{x, \text{pa}(x)\})$  by the algorithm. Because  $\sum_x p_x = 1$ , it follows in particular that the algorithm's output is a probability distribution. The remaining aspects of the RCAR condition are now easy to verify.

## References

1. Barlow, R.E., Bartholomew, D.J., Bremner, J.M., Brunk, H.D.: *Statistical Inference Under Order Restrictions: The Theory and Application of Isotonic Regression*. Wiley, New York (1972)
2. Boyd, S., Vandenberghe, L.: *Convex Optimization*. Cambridge University Press, Cambridge (2004)
3. Couso, I., Dubois, D.: A general framework for maximizing likelihood under incomplete data. *Int. J. Approximate Reasoning* **93**, 238–260 (2018)
4. Davies, P.L., Kovac, A.: Local extremes, runs, strings and multiresolution. *Ann. Stat.* **29**, 1–65 (2001)
5. Gill, R.D., Grünwald, P.D.: An algorithmic and a geometric characterization of coarsening at random. *Ann. Stat.* **36**, 2409–2422 (2008)
6. Grünwald, P.D., Dawid, A.P.: Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *Ann. Stat.* **32**, 1367–1433 (2004)
7. Grünwald, P.D., Halpern, J.Y.: Updating probabilities. *J. Artif. Intell. Res.* **19**, 243–278 (2003)
8. Guillaume, R., Couso, I., Dubois, D.: Maximum likelihood with coarse data based on robust optimisation. In: Antonucci, A., Corani, G., Couso, I., Destercke, S. (eds.) *Proceedings of the Tenth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA)*, pp. 169–180 (2017)
9. Heitjan, D.F., Rubin, D.B.: Ignorability and coarse data. *Ann. Stat.* **19**, 2244–2253 (1991)
10. Hüllermeier, E., Cheng, W.: Superset learning based on generalized loss minimization. In: Appice, A., Rodrigues, P.P., Santos Costa, V., Gama, J., Jorge, A., Soares, C. (eds.) *ECML PKDD 2015. LNCS (LNAI)*, vol. 9285, pp. 260–275. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-23525-7\\_16](https://doi.org/10.1007/978-3-319-23525-7_16)
11. Mammen, E., van de Geer, S.: Locally adaptive regression splines. *Ann. Stat.* **25**, 387–413 (1997)
12. Olariu, S., Overstreet, C.M., Wen, Z.: A mergeable double-ended priority queue. *Comput. J.* **31**(5), 423–427 (1991)
13. Schrijver, A.: *Combinatorial Optimization: Polyhedra and Efficiency*, vol. 24. Springer, Heidelberg (2003)
14. Selvin, S.: A problem in probability. *Am. Stat.* **29**, 67 (1975). Letter to the editor

15. Van Ommen, T.: Better predictions when models are wrong or underspecified. Ph.D. thesis, Mathematical Institute, Faculty of Science, Leiden (2015)
16. Van Ommen, T., Koolen, W.M., Feenstra, T.E., Grünwald, P.D.: Robust probability updating. *Int. J. Approximate Reasoning* **74**, 30–57 (2016)
17. Van Ommen, T.: Computing minimax decisions with incomplete observations. In: Antonucci, A., Corani, G., Couso, I., Destercke, S. (eds.) *Proceedings of the Tenth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA)*, pp. 358–369 (2017)