Learning Monotone Partitions of Partially-Ordered Domains (Work in Progress)

Oded Maler

Learning Monotone Partitions
of Partially-Ordered Domains

(Work in Progress)

Oded Maler

VERIMAG
CNRS and Univ. of Grenoble-Alpes
France
oded.maler@univ-grenoble-alpes.fr

July 4, 2017

Abstract

We present an algorithm for learning the boundary between an upward-closed set $\overline{X}$ and its downward-closed complement. The algorithm selects sampling points for which it submits membership queries $x \in \overline{X}$. Based on the answers and relying on monotonicity, it constructs an approximation of the boundary. The algorithm generalizes binary search on the continuum from one-dimensional (and linearly-ordered) domains to multi-dimensional (and partially-ordered) ones. Applications include the approximation of Pareto fronts in multi-criteria optimization and parameter synthesis for predicates where the influence of parameters is monotone.

1 Introduction and Motivation

Let X be a bounded and partially ordered set that we consider from now on to be $[0,1]^n$. A subset $\overline{X}$ of X is *upward closed* in X if

$$\forall x, x' \in X \; (x \in \overline{X} \wedge x' \geq x) \rightarrow x' \in \overline{X}.$$

Naturally, the complement of $\overline{X}$, $\underline{X} = X - \overline{X}$ is downward closed, and we use the term *monotone bi-partition* (or simply partition) for the pair $M = (\underline{X}, \overline{X})$. We do not have an explicit representation of M and we want to approximate it based on queries to a membership oracle which can answer for every $x \in X$ whether $x \in \overline{X}$.

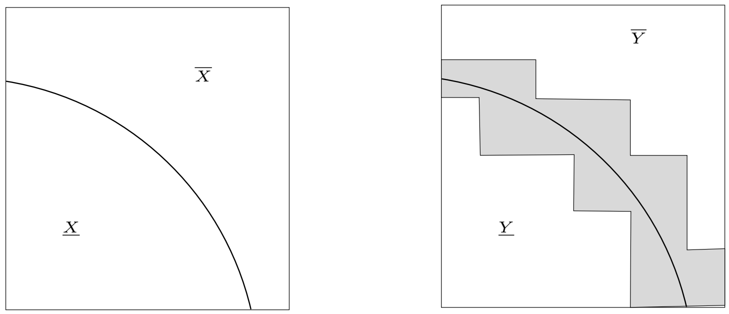

Figure 1: A monotone partition and its approximation.

Based on this information we construct an approximation of M by a pair of sets, $(\underline{Y}, \overline{Y})$ being, respectively, a downward-closed subset of $\underline{X}$ and an upward-closed subset of $\overline{X}$, see Figure 1. This approximation, conservative in both directions, says nothing about points residing in the gap between $\underline{Y}$ and $\overline{Y}$. This gap can be viewed as an over-approximation of $bd(M)$, the boundary between the two sets. There are two degenerate cases of monotone partitions, $(X, \emptyset)$ and $(\emptyset, X)$ that we ignore from now on, and thus assume that $\mathbf{0} \in \underline{X}$ and $\mathbf{1} \in \overline{X}$, where $\mathbf{r}$ denotes $(r, \ldots, r)$. We adopt the conventions that $bd(M)$ belongs to $\overline{X}$.

Before presenting the algorithmic solution that we offer to the problem, let us discuss some motivations. To start with, the problem is interesting for its own sake as a neat high-dimensional generalization of the problem of locating a boundary point that splits a straight line into two intervals. This problem is solved typically using binary (dichotomic) search, and indeed, the essence of our approach is in embedding binary search in higher dimension.

One major motivation comes from the domain of *multi-criteria optimization* where solutions are evaluated according to several criteria and the cost of a solution can be viewed as a point in a multi-dimensional cost space X. The optimal cost of such optimization problems is rarely a single point but rather a set of incomparable points also, known as the *Pareto front* of the problem. It consists of solutions that cannot be improved in one dimension without being worsened in another. Under certain assumptions, the Pareto front can be viewed as the boundary of a monotone partition. For a minimization problem, $\underline{X}$ corresponds to infeasible costs and $\overline{X}$ represents the feasible costs. The Pareto front is the set $bd(M) = \min(\overline{X})$ and the approximation that is provided is $\min(\overline{Y})$. In [2] we developed a procedure for computing such an approximation using a variant of binary search that submits queries to a constraint solver concerning the existence of solutions of a given cost x. The costs used in the queries were selected in order to reduce the distance between the boundaries of $\overline{Y}$ and $\underline{Y}$ and improve approximation quality. The present algorithm provides an alternative (and hopefully more efficient) way to approximate Pareto fronts.

Another motivation comes from some classes of parametric identification problems. Consider a parameterized family of predicates/constraints $\{\varphi_p\}$ where p is a vector of parameters ranging over some parameter space. Given an element u from the domain of the predicates, we would like to know the range of parameters p such that $\varphi_p(u)$ holds. We say that a parameter p has a fixed (positive or negative) polarity if increasing its value will have a monotone effect on the set of elements that satisfy it. For example if a parameter p appears in a parameterized predicate $u \leq p$, then for any $p' > p$ and any u, $\varphi_p(u)$ implies $\varphi_{p'}(u)$. When no parameter appears in two constraints in opposing sides of an inequality, and after some pre-processing, the set of parameters that lead to satisfaction is upward closed. Its set of minimal elements indicates the set of tightest parameters that lead to satisfaction of $\varphi_p(u)$, which is a valuable information about u. In [1] we explored the idea for the domain of real-valued signals $u(t)$ and temporal formulas such as $\exists t < p_1\ u(t) < p_2$.

2 Binary Search in One Dimension

Our major tool is classical binary search over one-dimensional and totally-ordered domains, where a partition of $[0, 1]$ is of the form $M = ([0, z), [z, 1])$ for some $0 < z < 1$. The outcome of the search procedure is a pair of numbers $\underline{y}$ and $\overline{y}$ such that $\underline{y} < z < \overline{y}$, which implies a partition approximation $M' = ([0, \underline{y}), [\overline{y}, 1])$. The quality of M' is measured by the size of the gap $\overline{y} - \underline{y}$, which can be made as small as needed by running more steps. Note that in one dimension, $\overline{y} - \underline{y}$ is both the volume of $[\underline{y}, \overline{y}]$ and its diameter. We are going to apply binary search to straight lines of arbitrary position and arbitrary positive orientation inside high-dimensional X, hence we formulate it in terms that will facilitate its application in this context.

Definition 1 (Line Segments in High-Dimension) *The line segment connecting two points $\underline{x} < \overline{x} \in X = [0, 1]^n$ is their convex hull*

$$\langle \underline{x}, \overline{x} \rangle = \{(1 - \lambda)\underline{x} + \lambda \overline{x} : \lambda \in [0, 1]\}.$$

The segment inherits a total order from $[0, 1]$: $x \leq x'$ whenever $\lambda \leq \lambda'$.

The input to the binary search procedure, written in Algorithm 1, is a line segment ℓ and an oracle for a monotone partition $M = (\underline{\ell}, \overline{\ell}) = (\langle \underline{x}, z \rangle, \langle z, \overline{x} \rangle)$, $\underline{x} < z < \overline{x}$. The output is a sub-segment $\langle \underline{y}, \overline{y} \rangle$ containing the boundary point z.

The procedure is parameterized by an error bound $\epsilon \geq 0$, with $\epsilon = 0$ representing an ideal variant of the algorithm that runs indefinitely and finds the exact boundary point. Although realizable only in the limit, it is sometimes convenient to speak in terms of this variant. Figure 2 illustrates several steps of the algorithm.

Algorithm 1 One dimensional binary search: $search(\langle \underline{x}, \overline{x} \rangle, \epsilon)$

Input: A line segment $\ell = \langle \underline{x}, \overline{x} \rangle$, a monotone partition $M = (\underline{\ell}, \overline{\ell})$ accessible via an oracle $member()$ for membership in $\overline{\ell}$ and an error bound $\epsilon \geq 0$.
Output: A line segment $\langle \overline{y}, \underline{y} \rangle$ containing $bd(M)$ such that $\overline{y} - \underline{y} \leq \epsilon$.
$\langle \underline{y}, \overline{y} \rangle = \langle \underline{x}, \overline{x} \rangle$
while $\overline{y} - \underline{y} \geq \epsilon$ **do**
  $y = (\underline{y} + \overline{y})/2$
  if $member(y)$ **then**
    $\langle \underline{y}, \overline{y} \rangle = \langle \underline{y}, y \rangle$ ▷ left sub-interval
  else
    $\langle \underline{y}, \overline{y} \rangle = \langle y, \overline{y} \rangle$ ▷ right sub-interval
  end if
end while
return $\{\langle \underline{y}, \overline{y} \rangle\}$

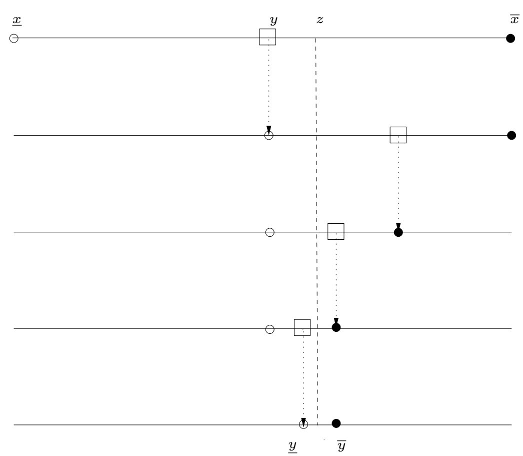

Figure 2: Binary search and the successive reduction of the uncertainty interval.

3 Monotone Partitions in High Dimension

The following definitions are commonly used in multi-criteria optimization and in partially-ordered sets in general.

Definition 2 (Domination and Incomparability) *Let $x = (x_1, \ldots, x_n)$ and $x' = (x'_1, \ldots, x'_n)$ be two points. Then*

1. *$x \leq_i x'$ if $x_i \leq x'_i$; $x <_i x'$ if $x_i < x'_i$;*
2. *$x \leq x'$ if $x \leq_i x'$ for every i;*
3. *$x < x'$ if $x \leq x'$ and $x <_i x'$ for some i. In this case we say that x dominates x';*
4. *$x||x'$ if $x \not\leq x'$ and $x' \not\leq x$, which means that $x <_i x'$ and $x' <_j x'$ for some i and j. In this case we say that x and x' are incomparable.*

Any two points $\underline{x} < \overline{x}$ define a rectangle $\lfloor \underline{x}, \overline{x} \rceil = \{x : \underline{x} \leq x \leq \overline{x}\}$ for which they are, respectively, the minimal and maximal corners, as well as the endpoints of the diagonal $\langle \underline{x}, \overline{x} \rangle$. A point x defines various rectangles consisting of points with which it is in certain order relations. Similar relations can be associated with a rectangle $\lfloor \underline{x}, \overline{x} \rceil$.

Definition 3 (Rectangular Half-Space) *Let $i \in \{1, \ldots, n\}$ be a dimension.*

- *The orthogonal i-half-spaces associated with a point $x \in X$ are*

$$C_{i,0}(x) = \{x' \in X : x' \leq_i x\},\ C_{i,1}(x) = \{x' \in X : x' \geq_i x\}.$$

- *The orthogonal i-half-spaces associated with a rectangle $\lfloor \underline{x}, \overline{x} \rceil \subseteq X$ are*

$$C_{i,0}(\lfloor \underline{x}, \overline{x} \rceil) = \{x' \in X : x' \leq_i \overline{x}\},\ C_{i,1}(\lfloor \underline{x}, \overline{x} \rceil) = \{x' \in X : x' \geq_i \underline{x}\}.$$

Observe that for a point, the half-spaces $C_{i,0}(x)$ and $C_{i,1}(x)$ form a partition while for a rectangle $C_{i,0}(\lfloor \underline{x}, \overline{x} \rceil)$ and $C_{i,1}(\lfloor \underline{x}, \overline{x} \rceil$ overlap over the interval $[\underline{x}_i, \overline{x}_i]$. They can be defined, alternatively, as all the points x' for which there exists $x \in \lfloor \underline{x}, \overline{x} \rceil$ such that $x' \leq_i x$ (resp. $x \geq_i x'$).

Definition 4 (Rectangular Cones) *Let $\alpha \in \{0, 1\}^n$ be a Boolean vector.*

- *The rectangular α-cone induced by a point x is*

$$B_\alpha(x) = \bigcap_{i=1}^{n} C_{i,\alpha_i}(x).$$

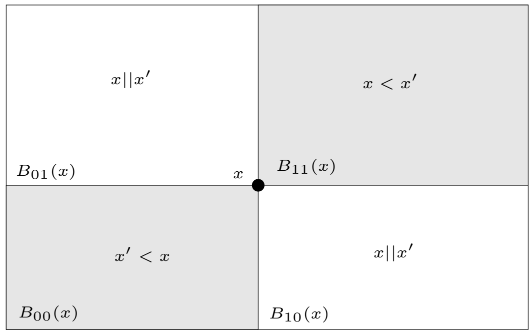

Figure 3: Rectangular cones in dimension 2.

- *The rectangular α-cone induced by a rectangle $\lfloor \underline{x}, \overline{x} \rceil$ is*

$$B_\alpha(\lfloor \underline{x}, \overline{x} \rceil) = \bigcap_{i=1}^{n} C_{i,\alpha_i}(\lfloor \underline{x}, \overline{x} \rceil).$$

The rectangular cones of x partition X into 2^n boxes and x is a corner of each. In particular, $B_{\mathbf{0}}(x) = \lfloor \mathbf{0}, x \rceil$ and $B_{\mathbf{1}}(x) = \lfloor x, \mathbf{1} \rceil$ are, the downward and upward cones of x, consisting, respectively, of points below and above x. The set of all other $2^n - 2$ cones, denoted by $I(x)$, contains rectangles consisting of points incomparable to x. These notions are illustrated for $n = 2$ in Figure 3. Naturally, the cones associated with a rectangle do not form a partition, and only $B_{\mathbf{0}}(\lfloor \underline{x}, \overline{x} \rceil) = B_{\mathbf{0}}(\underline{x})$ and $B_{\mathbf{1}}(\lfloor \underline{x}, \overline{x} \rceil) = B_{\mathbf{1}}(\overline{x})$ are separated from the other cones. We use $I(\lfloor \underline{x}, \overline{x} \rceil)$ for the incomparable cones.

The multi-dimensional algorithm presented in the sequel is based on the observation that any line ℓ of a positive slope inside a rectangle $\lfloor \underline{x}, \overline{x} \rceil$ that admits a monotone partition M, intersects $bd(M)$ at most once. In particular, the diagonal $\ell = \langle \underline{x}, \overline{x} \rangle$ of the rectangle is guaranteed to intersect $bd(M)$. Hence ℓ admits by itself a monotone partition $(\underline{\ell}, \overline{\ell})$ that can be subject to Algorithm 1 to obtain an approximation of $bd(M) \cap \ell$.

Figure 4 illustrates the interaction between the result of the one-dimensional search process on the diagonal and the approximation of the higher-dimensional partition. Initially both $\underline{Y}$ and $\overline{Y}$ are set to $\emptyset$ and their complement, the over-approximation of $bd(M)$, is the whole domain X. Figure 4-(a) shows the outcome of running the ideal version of Algorithm 1 which finds the boundary point y. In this case the upward cone of y is added to $\overline{X}$ and the downward cone is added to $\underline{Y}$. The boundary approximation is thus refined to become their complement, the union $B_{01}(y) \cup B_{10}(y)$ of the rectangles incomparable to y.

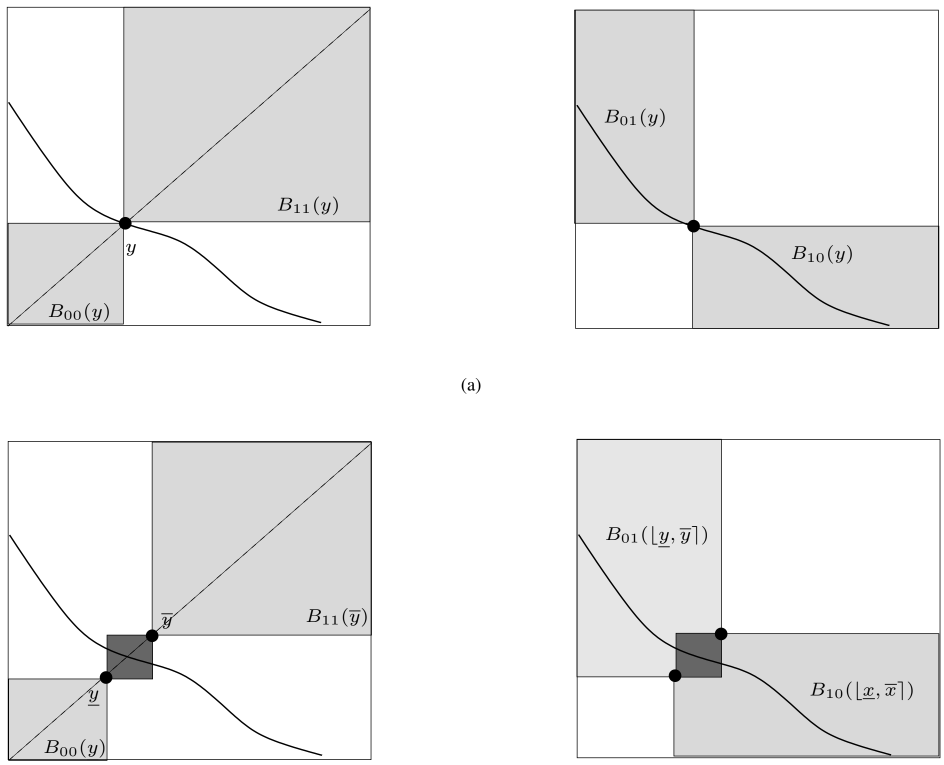

Figure 4: (a) The effect of finding the exact intersection of the diagonal with the boundary; (b) The effect of finding an interval approximation of that intersection.

The situation with the non-ideal variant of the search algorithm is a bit more involved qualitatively, but since ϵ can be easily made small, it does not make a big quantitative difference. Figure 4-(b) shows the outcome of a search process that approximates the boundary of the one-dimensional partition by $\langle \underline{y}, \overline{y} \rangle$. In this case only $B_{11}(\overline{y})$ and $B_{00}(\underline{y})$ can be classified with certainty while the partition of $\lfloor \underline{y}, \overline{y} \rceil$ between $\underline{X}$ and $\overline{X}$ is unknown. In order to guarantee a safe approximation of the boundary, the points in $\lfloor \underline{y}, \overline{y} \rceil$ are treated as incomparable. The boundary approximation is refined into the union of the two overlapping rectangles $B_{01}(\lfloor \underline{y}, \overline{y} \rceil)$ and $B_{10}(\lfloor \underline{y}, \overline{y} \rceil)$.

The whole procedure for learning/approximating a monotone partition is written down in Algorithm 2. It maintains at any moment the current approximation $(\underline{Y}, \overline{Y})$ of the partition as well s its complement represented as a list L of rectangles

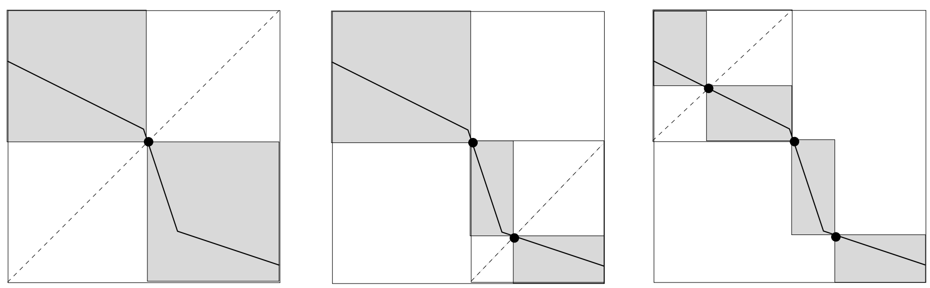

Figure 5: Successive approximation of the partition boundary by running binary search on diagonals of incomparable boxes.

whose union constitutes an over-approximation of the boundary. For efficiency reasons, L is maintained in a decreasing size order. We successively take the largest rectangle from L, run binary search on its diagonal and refine it until some stopping criterion on the size of the boundary approximation (for example total volume) is met. A Some steps of the algorithm are illustrated in Figure 5.

Algorithm 2 Approximating a monotone partition (and its boundary) by unions of rectangular cones.

Input: A rectangle X, a partition $M = (\underline{X}, \overline{X})$ accessed by a membership oracle for $\overline{X}$ and an error bound δ.
Output: An approximation $M' = (\underline{Y}, \overline{Y})$ of M and an approximation L of the boundary $bd(M)$ such that $|L| \leq \delta$. All sets are represented by unions of rectangles.
$L = \{X\}$; $(\underline{Y}, \overline{Y}) = (\emptyset, \emptyset)$ ▷ initialization
repeat
    pop first $\lfloor \underline{x}, \overline{x} \rceil \in L$ ▷ take the largest rectangle from the boundary approximation
    $\langle \underline{y}, \overline{y} \rangle = search(\langle \underline{x}, \overline{x} \rangle, \epsilon)$ ▷ run binary search on the diagonal
    $\underline{Y} = \underline{Y} \cup \{B_{\mathbf{0}}(\underline{y})\}$ ▷ add backward cone
    $\overline{Y} = \overline{Y} \cup \{B_{\mathbf{1}}(\overline{y})\}$ ▷ add forward cone
    $L = L \cup I(\lfloor \underline{x}, \overline{x} \rceil)$ ▷ insert incomparable rectangles to L
until $|L| \leq \delta$

4 Current Status

Having presented the algorithm, what remains is to evaluate its performance, both empirically and theoretically. The algorithm has been implemented by Marcell Vazquez-Chanlatte, and it will hopefully be used in the future for systematic evaluation. Preliminary ideas on worst-case complexity were proposed by Nicolas Basset and Eugene Asarin. Other useful comments were made by Alexey Bakhirkin and Dogan Ulus.

References

[1] Eugene Asarin, Alexandre Donzé, Oded Maler, and Dejan Nickovic. Parametric identification of temporal properties. In *RV*, pages 147–160, 2011.

[2] Julien Legriel, Colas Le Guernic, Scott Cotton, and Oded Maler. Approximating the Pareto front of multi-criteria optimization problems. In *TACAS*, pages 69–83, 2010.