Diverse Voronoi Partitions of 1D Colored Points*

Marc van Kreveld[1], **Bettina Speckmann**[2], **and Jérôme Urhausen**[1]

1 **Dep. of Information and Computing Sciences, Utrecht University**
`{m.j.vankreveld|j.e.urhausen}@uu.nl`

2 **Dep. of Mathematics and Computer Science, TU Eindhoven**
`b.speckmann@tue.nl`

Abstract

We introduce the diverse Voronoi partition problem: for a given set of colored points and a number k, determine a set of k point sites so that the Voronoi cells of these sites contain as many colors as possible. We show that the problem is already NP-complete for points colored in four colors on a line, and give polynomial-time algorithms for a few special cases.

1 Introduction

Inspired by recent research on algorithmic fairness and similar concepts [1, 8], we study a problem involving diversity and representation. Imagine that a set of objects is represented by points in a space, and different classes are represented by colors. How can we represent all of the colored points by a smaller set of points, each of which represents many colors? We call the representing points *sites*. To formulate such problems, we need to specify when a site represents a colored point. The most obvious choice is by distance: a colored point will always be represented by the nearest site. A site is *diverse* if it represents many colors, in particular, the diversity score of a site is the number of colors it represents. There are different options to combine the diversity scores of sites into a global diversity score. We choose the sum measure. This leads to the following problem:

Diverse Voronoi Partition (DVP)
Input: Set P containing n points of h different colors and a number $k \in \mathbb{N}$.
Question: Determine a point set $S = \{s_1, \ldots, s_k\}$ that maximizes $\sum_{i=1}^{k} c_i$, where, in the Voronoi Diagram of S, c_i is the number of colors present in the cell of s_i.

For $i \in \{1, \ldots, h\}$, we define $P_i \subseteq P$ as the set of points of color i. One way to view the goal is to find a set of sites S, which represents each set P_i. The number of colors c_i in the cell of s_i is called the *score* of that cell, and $\sum c_i$ is the *score* of the Voronoi partition. The sites in Figure 1 for example have a score of 9.

A lot of related research exists in computational geometry. Firstly, various problems on colored points have been studied, see for example Kaneko and Kano [9]. In some cases the problems concern partitioning. For example, Dumitrescu and Pach [6] study partitioning multi-colored point sets into uni-colored subsets by convex cells, Majumder et al. [10] consider the same but partition with lines, and Bespamyatnikh et al. [4] consider partitioning a red-blue point set into convex cells so that each cell has the same number of red points and the same number of blue points (extensions were given in [2, 3, 5]).

Secondly, our problem is a type of clustering problem reminiscent of k-means clustering, where a representation of multiple points by a single point is also used, following a nearest

* Research on the topic of this paper was initiated at the 4th Workshop on Applied Geometric Algorithms (AGA 2018) in Langbroek, The Netherlands, supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 639.023.208. The first and third authors are supported by the NWO TOP grant no. 612.001.651.

36th European Workshop on Computational Geometry, Würzburg, Germany, March 16–18, 2020.
This is an extended abstract of a presentation given at EuroCG'20. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

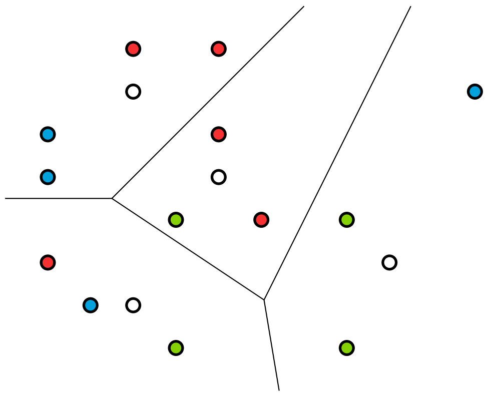

Figure 1 The sites (the white disks) induce a Voronoi diagram that determines which points a site represents. The bottom-left cell has a score of 3; the other three cells have a score of 2 each.

neighbor rule. While k-means clustering aims to minimize the sum of squared distances to the nearest representative, our version has colored points and aims to maximize the number of colors close to each representative.

Thirdly, our problem bears some resemblance to multi-criteria facility location [7] where multiple facilities are placed.

We restrict ourselves to points and sites on a line. We prove that even then, and with only four colors, the problem is NP-hard. In contrast, dropping the condition that the k cells be Voronoi intervals of points on the real line makes the problem solvable in polynomial time by dynamic programming, for any number of colors. We consider special cases where optimally diverse Voronoi partitions can be found in polynomial time. One such case is where a discrete set of m candidate locations is given on which the k sites of S should lie. A second case is where we have exactly n/h points per color and we want to know if an optimal solution with $k = n/h$ sites in S exists of score n; we call this a *perfect partition*.

2 Preliminaries

In this section we explore the difference between a Voronoi partition and any partition of a real line into intervals.

Assume a set P of n colored points is given. Assume that we roughly know where the interval ends should be to maximize the score, when placing k points. These interval ends, or *boundaries*, can be specified by intervals themselves, called *b-intervals*.

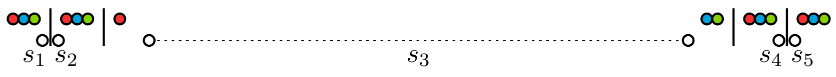

Figure 2 15 colored points that admit a perfect partition but not a perfect Voronoi partition.

Figure 2 shows a situation with 15 colored points, five in each of three colors. It is easy to see that a partition exists into five intervals such that the score is 15, the maximum possible. However, to find a Voronoi partition, we need to place five sites such that the boundaries are at the desired places, which are the b-intervals. A careful inspection shows that we cannot place five sites to realize a score of 15. The middle site must be sufficiently far to the left to ensure that the second boundary is correctly placed, and also sufficiently far to the right to ensure that the third boundary is correctly placed. We cannot move the sites s_1, s_2, s_4, s_5 enough to realize this.

When we need to place sites $s_1, \ldots, s_k$ so that the Voronoi cell boundaries lie exactly in given b-intervals, we have a set of constraints to satisfy. Let b_i be the midpoint of s_i and s_{i+1}, for $1 \leq i < k$. Then $s_1 \leq b_1 \leq s_2 \leq \cdots b_{i-k} \leq s_k$, and $b_i = (s_i + s_{i+1})/2$.

To ensure that the Voronoi cell boundaries b_i lie inside their respective b-intervals, we use another $2k-2$ linear inequalities. Altogether, we have set up a system of $5k-5$ linear inequalities whose solution—if it exists—gives a Voronoi partition.

A specific case where this approach works is the following. Assume we have n points, equally many in each of the h colors. The problem is to place $k = n/h$ sites to answer the question whether the total score of n can be realized. We solve this problem as follows. We first check if a partition exists, which is only the case when the points come in n/h groups of points with h different colors. It is clear where the boundaries should be, and we can set up the system of linear inequalities in $O(n)$ time. Then we solve the linear program in time polynomial in $k = n/h$. If it is infeasible, there is no Voronoi partition that is perfect.

3 Algorithm for Discrete Site Locations

Here we assume that sites can only be placed at a finite set M of prespecified positions. We show that with dynamic programming, we can develop a polynomial-time algorithm.

The dynamic program works its way from left to right, using the fact that in a placement of the first i sites, we use an optimal placement of the first $i-1$ sites. However, the Voronoi boundaries between sites are determined by the last two sites, so our recurrence for an optimal solution has parameters for the last two sites. Furthermore, since we do not know the score for the i-th site, since its right boundary has not been fixed, we define maximum total score for the first $i-1$ sites for locations of the $(i-1)$-th and i-th sites.

Consider the function $f : \{2, \ldots, k\} \times M \times M \to \mathbb{N}$. For $i \in \{2, \ldots, k\}$ and $u, v \in M$, with $u < v$, let

$$f(i,u,v) = \max_{s_1,\ldots,s_{i-2} \in M} \Big(\sum_{j=1}^{i-1} c_j \mid s_1 < \cdots < s_i,\ s_{i-1} = u,\ s_i = v\Big)$$

be the best possible score of the first $i-1$ cells while fixing $s_{i-1} = u$ and $s_i = v$.

We further define the function $g : (M \cup \{-\infty\}) \times M \times (M \cup \{+\infty\}) \to \mathbb{N}$, where $g(a,b,c)$ is the number of colors present in the cell corresponding to the site b, where a and c are its left and right neighboring sites.

For $a, b \in M$, we have $f(2,a,b) = g(-\infty,a,b)$. Additionally we have the recursive definition:

$$\forall i \in \{3, \ldots, k\}, \forall b, c \in M, \text{ with } b < c,\ f(i,b,c) = \max_{a \in M, a<b} \big(f(i-1,a,b) + g(a,b,c)\big)\ . \quad (1)$$

In order to find the best sites overall we determine $\max_{a,b \in M, a<b}(f(k,a,b) + g(a,b,+\infty))$, which includes the number of colors in the k-th cell in the second term. The dynamic program uses a table of size $O(km^2)$, where $m = |M|$, and filling an entry requires optimizing over m choices. Since the function g can be evaluated trivially in $O(n)$ time, we immediately get an $O(km^3n)$ time algorithm. The positions of the sites can then be obtained by backtracking as usual for dynamic programming algorithms.

We can improve the running time by preprocessing, to be able to evaluate g faster. Note that there are only $O(m^3)$ different values for g, whereas the function is evaluated $O(km^3)$

times. We will show how to precompute all values of g, so that a lookup during the main algorithm evaluates g in only $O(1)$ time:

1. For each $p \in P$, make a sorted list of the leftmost points right of it, one per color.
2. For each $b \in M$, make a sorted list of the boundaries $(b + c)/2$ with $c \in M$, $c > b$ in sorted order. This list has at most $m - 1$ entries (the options for c).
3. For each $a, b \in M$, $a < b$, store the rightmost point $p(a, b) \in P$ left of $(a + b)/2$.

This information can be computed easily in $O(m^2+nh)$ time. Then, for each $a, b \in M$, we access $p(a, b)$ to get access to the sorted list of points with unique colors, stored with $p(a, b)$. We simultaneously scan the sorted list of colored points and the sorted list of boundaries $(b + c)/2$ to fill in all values $g(a, b, \cdot)$ in $O(m + h)$ time.

The running time of the whole algorithm becomes $O(km^3 + m^2h + nh)$.

4 Deciding DVP is NP-complete

We prove that the decision version of DVP is NP-complete. The decision version, referred to as D-DVP, receives an extra parameter z and asks if a diversity score of at least z can be realized with a Voronoi partition using k points.

We start by proving containment in NP. For a given instance of D-DVP, there are only exponentially many partitions into subsets, each defined by $k - 1$ b-intervals, and we can test them all non-deterministically in polynomial time by linear programming to decide if they allow Voronoi partitions of k sites (see Section 2).

In order to prove hardness we reduce from Subset Sum.

Subset Sum
Input: A set $A = \{a_1, \ldots, a_r\}$ of integers and an integer b.
Question: Is there a subset $A' \subseteq A$ such that $\sum_{a_j \in A'} a_j = b$?

Before starting with the reduction, we define a few terms. A point $p \in P$ is *represented* by a site $s \in S$ if s is the site closest to p. We say that for each color $i \in \{1, \ldots, h\}$, a point $p \in P_i$ is *scored* if each other point p' of the same color that is represented by the same site is to the right of p. That is, for each site only the leftmost point of each color that it represents is scored. It follows that finding the optimal S is equal to maximizing the number of scored points. A point is *unscored* if it is not scored. Our reduction uses only four colors, so we will define point sets P_1, P_2, P_3, P_4 from an instance of Subset Sum.

Let $0 < \delta \ll 1$ be a small real and let $0 < \varepsilon \ll \delta$ be an even smaller real. We can take $\delta = 1/n^2$ and $\varepsilon = 1/n^4$. Later we can multiply each coordinate by n^4 and thus obtain a set of integer positions with polynomial size. Let $a = \sum_{i=1}^{r} a_i$ be the sum of all input values.

We describe P from left to right. First, there is a starting gadget H of six points. Then we have a gadget D^j for each a_j, consisting of eight points (the gadgets can be in any order). Next, we have a subset sum gadget E of two points to represent b, and finally we have an ending gadget G of six points. $P = H \cup D^1 \cup \cdots \cup D^r \cup E \cup G$. Figure 3 shows an example for $A = \{1, 2\}$, so $P = H \cup D^1 \cup D^2 \cup E \cup G$.

To start the construction we define a set H of six points in two colors. We set $H_1 = \{-2\delta, -\delta, 0\} \subset P_1$ and $H_2 = \{-2\delta - \varepsilon, -\delta - \varepsilon, -\varepsilon\} \subset P_2$. The set H thus forms three groups of two points of different colors. We can only score all points in H with three sites s_{-2}, s_{-1}, s_0 if we have $-\delta < s_0 < 2\delta - 2\varepsilon$. So, in order to score all six points with three sites, the rightmost of those sites, s_0, needs to be close to zero.

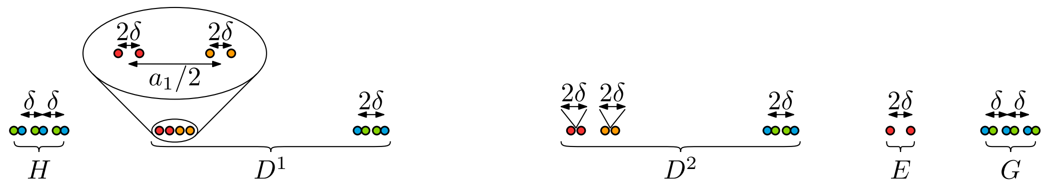

Figure 3 The reduction from SUBSET SUM to D-DVP illustrated. We have $a_1 = 1$, $a_2 = 2$ and $b = 2$. The points of P_1, P_2, P_3, and P_4 are blue, green, red, and yellow, respectively. Any two touching points are considered to be at ε distance. Note that δ is not drawn to scale, for clarity.

For each $a_j \in A$ we create a set D^j of eight points that will somehow encode whether a_j is chosen in the subset A' or not. Let $D^j = D^j_1 \cup D^j_2 \cup D^j_3 \cup D^j_4$, with $D^j_i \subset P_i$ (the points in D^j_i have color i). Let $D^j_1 = \{(4j-1)a-\delta, (4j-1)a+\delta\}$, $D^j_2 = \{(4j-1)a-\delta+\varepsilon, (4j-1)a+\delta-\varepsilon\}$, $D^j_3 = \{(4j-3)a-\delta, (4j-3)a+\delta\}$ and $D^j_4 = \{(4j-3)a+a_j/2-\delta, (4j-3)a+a_j/2+\delta\}$. The distances between D^j_3 and D^j_4 are roughly $a_j/2$.

We define a set $E \subset P$ that encodes that we want the subset sum to be b. We define $E \subset P_3$ with $E = \{4ra + (a-b)/2 - \delta, 4ra + (a-b)/2 + \delta\}$.

Finally we define a set G of six points to end the construction. It is similar to H. It can only be scored fully by three sites if the leftmost of the sites is close to $(4r+1)a$. We set $G_1 = \{(4r+1)a, (4r+1)a+\delta, (4r+1)a+2\delta\} \subset P_1$ and $G_2 = \{(4r+1)a+\varepsilon, (4r+1)a+\delta+\varepsilon, (4r+1)a+2\delta+\varepsilon\} \subset P_2$.

The instance of D-DVP has $n = 8r + 14$ points and asks to place $k = 2r + 6$ sites to realize a score of $z = 7r + 14$, which can be achieved if and only if the corresponding SUBSET SUM instance has a solution. Intuitively, we want to use the sites to create boundaries that separate the first three pairs of points, the last three pairs of points, either D^j_3 or D^j_4, and also E. Separating D^j_3 corresponds to not choosing a_j in a subset and separating D^j_4 corresponds to choosing a_j. If we choose the correct a_j, the boundary between the last site chosen for D^r and the first site chosen for G will magically separate the points in E, due to its careful placement. Then, only one point of each D^j is not scored. An example can be seen in Figure 4.

The proof of correctness essentially shows that there are no other options: the SUBSET SUM instance has a solution if and only if the DVP instance can score $7r + 14$. For details we refer to the full version.

References

1 M. Abbasi, S. A. Friedler, C. Scheidegger, and S. Venkatasubramanian. Fairness in representation: quantifying stereotyping as a representational harm. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 801–809. SIAM, 2019.

2 S. Bereg, P. Bose, and D. Kirkpatrick. Equitable subdivisions within polygonal regions. *Computational Geometry*, 34(1):20–27, 2006.

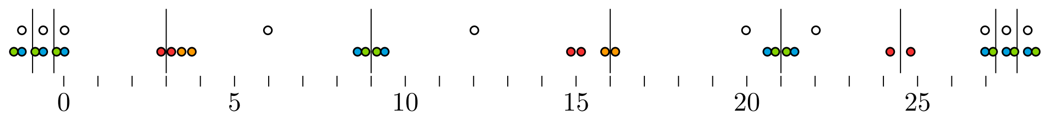

Figure 4 An induced D-DVP instance is illustrated. Here we have $a_1 = 1$, $a_2 = 2$ and $b = 2$. The sites and boundaries for a score of $7r + 14$ are indicated.

3 S. Bereg, F. Hurtado, M. Kano, M. Korman, D. Lara, C. Seara, R. I. Silveira, J. Urrutia, and K. Verbeek. Balanced partitions of 3-colored geometric sets in the plane. *Discrete Applied Mathematics*, 181:21–32, 2015.

4 S. Bespamyatnikh, D. Kirkpatrick, and J. Snoeyink. Generalizing ham sandwich cuts to equitable subdivisions. *Discrete & Computational Geometry*, 24(4):605–622, 2000.

5 F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii. Fair clustering through fairlets. In *Advances in Neural Information Processing Systems*, pages 5029–5037, 2017.

6 A. Dumitrescu and J. Pach. Partitioning colored point sets into monochromatic parts. *International Journal of Computational Geometry & Applications*, 12(05):401–412, 2002.

7 R. Z. Farahani, M. SteadieSeifi, and N. Asgari. Multiple criteria facility location problems: A survey. *Applied Mathematical Modelling*, 34(7):1689–1709, 2010.

8 V. Gupta, P. Nokhiz, C. D. Roy, and S. Venkatasubramanian. Equalizing recourse across groups. *arXiv preprint arXiv:1909.03166*, 2019.

9 A. Kaneko and M. Kano. Discrete geometry on red and blue points in the plane, a survey. In *Discrete and Computational Geometry, The Goodman-Pollack Festschrift*, pages 551–570. Springer, 2003.

10 S. Majumder, S. C. Nandy, and B. B. Bhattacharya. Separating multi-color points on a plane with fewest axis-parallel lines. *Fundamenta Informaticae*, 99(3):315–324, 2010.