mathematics

MDPI

Article

Unified Polynomial Dynamic Programming Algorithms for P-Center Variants in a 2D Pareto Front [†]

Nicolas Dupin [1,*], Frank Nielsen [2] and El-Ghazali Talbi [3]

1 Université Paris-Saclay, CNRS, Laboratoire Interdisciplinaire des Sciences du Numérique, 91400 Orsay, France
2 Sony Computer Science Laboratories Inc., Tokyo 141-0022, Japan; Frank.Nielsen@acm.org
3 CNRS UMR 9189-CRIStAL-Centre de Recherche en Informatique Signal et Automatique de Lille, Université Lille, F-59000 Lille, France; el-ghazali.talbi@univ-lille.fr
* Correspondence: nicolas.dupin@universite-paris-saclay.fr

† This paper is an extended version of our paper published in OLA 2020, International Conference in Optimization and Learning, Cadiz, Spain, 17–19 February 2020.

Abstract: With many efficient solutions for a multi-objective optimization problem, this paper aims to cluster the Pareto Front in a given number of clusters K and to detect isolated points. K-center problems and variants are investigated with a unified formulation considering the discrete and continuous versions, partial K-center problems, and their min-sum-K-radii variants. In dimension three (or upper), this induces NP-hard complexities. In the planar case, common optimality property is proven: non-nested optimal solutions exist. This induces a common dynamic programming algorithm running in polynomial time. Specific improvements hold for some variants, such as K-center problems and min-sum K-radii on a line. When applied to N points and allowing to uncover $M < N$ points, K-center and min-sum-K-radii variants are, respectively, solvable in $O(K(M+1)N \log N)$ and $O(K(M+1)N^2)$ time. Such complexity of results allows an efficient straightforward implementation. Parallel implementations can also be designed for a practical speed-up. Their application inside multi-objective heuristics is discussed to archive partial Pareto fronts, with a special interest in partial clustering variants.

Keywords: discrete optimization; operational research; computational geometry; complexity; algorithms; dynamic programming; clustering; k-center; p-center; sum-radii clustering; sum-diameter clustering; bi-objective optimization; Pareto Front; parallel programming

check for updates

Citation: Dupin, N.; Nielsen, F.; Talbi, E.-G. Unified Polynomial Dynamic Programming Algorithms for P-Center Variants in a 2D Pareto Front. *Mathematics* **2021**, *9*, 453. https://doi.org/10.3390/math9040453

Academic Editor: Frank Werner

Received: 21 December 2020
Accepted: 16 February 2021
Published: 23 February 2021

1. Introduction

This paper is motivated by real-world applications of multi-objective optimization (MOO). Some optimization problems are driven by more than one objective function, with conflicting optimization directions. For example, one may minimize financial costs, while maximizing the robustness to uncertainties or minimizing the environmental impact [1,2]. In such cases, higher levels of robustness or sustainability are likely to induce financial over-costs. Pareto dominance, preferring one solution to another if it is better for all the objectives, is a weak dominance rule. With conflicting objectives, several non-dominated points in the objective space can be generated, defining efficient solutions, which are the best compromises. A Pareto front (PF) is the projection in the objective space of the efficient solutions [3]. MOO approaches may generate large sets of efficient solutions using Pareto dominance [3]. Summarizing the shape of a PF may be required for presentation to decision makers. In such a context, clustering problems are useful to support decision making to present a view of a PF in clusters, the density of points in the cluster, or to select the most central cluster points as representative points. Note than similar problems are of interest for population MOO heuristics such as evolutionary algorithms to archive representative points of a partial Pareto fronts, or in selecting diversified efficient solutions to process mutation or cross-over operators [4,5].

With N points in a PF, one wishes to define $K \ll N$ clusters while minimizing the measure of dissimilarity. The K-center problems, both in the discrete and continuous versions, define the cluster costs in this paper, covering the PF with K identical balls while minimizing the radius of the balls used. By definition, the ball centers belong to the PF for the discrete K-center version, whereas the continuous version is similar to geometric covering problems, without any constraint for the localization of centers. Furthermore, sum-radii or sum-diameter are min-sum clustering variants, where the covering balls are not necessarily identical. For each variant, one can also consider partial clustering variants, where a given percentage (or number) of points can be ignored in the covering constraints, which is useful when modelling outliers in the data.

The K-center problems are NP-hard in the general case, [6] but also for the specific case in $\mathbb{R}^2$ using the Euclidean distance [7]. This implies that K-center problems in three-dimensional (3D) PF are also NP-hard, with the planar case being equivalent to an affine 3D PF. We consider the case of two-dimensional (2D) PF in this paper, focusing on the polynomial complexity results. It as an application to bi-objective optimization, the 3D PF and upper dimensions are shown as perspectives after this work. Note that 2D PF are a generalization of one-dimensional (1D) cases, where polynomial complexity results are known [8,9]. A preliminary work proved that K-center clustering variants in a 2D PF are solvable in polynomial time using a Dynamic Programming (DP) algorithm [10]. This paper improves these algorithms for these variants, with an extension to min-sum clustering variants, partial clustering, and Chebyshev and Minkowski distances. The properties of the DP algorithms are discussed for efficient implementation, including parallelization.

This paper is organized as follows. The considered problems are defined formally with unified notations in Section 2. In Section 3, related state-of-the-art elements are discussed. In Sections 4 and 5, intermediate results and specific complexity results for sub-cases are presented. In Section 6, a unified DP algorithm with a proven polynomial complexity is designed. In Section 7, specific improvements are presented. In Section 8, the implications and applications of the results of Sections 5–7 are discussed. In Section 9, our contributions are summarized, with a discussion of future research directions.

2. Problem Statement and Notation

In this paper, integer intervals are denoted as $[\![a,b]\!] = [a,b] \cap \mathbb{Z}$. Let $E = \{x_1, \dots, x_N\} = \{x_i\}_{i \in [\![1,N]\!]}$ a set of N elements of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \ \mathcal{I} \ x_j$ defining the binary relations $\mathcal{I}, \prec$ for all $y = (y^1, y^2), z = (z^1, z^2) \in \mathbb{R}^2$ with

$$y \prec z \iff y^1 < z^1 \text{ and } y^2 > z^2 \tag{1}$$

$$y \preccurlyeq z \iff y \prec z \text{ or } y = z \tag{2}$$

$$y \ \mathcal{I} \ z \iff y \prec z \text{ or } z \prec y \tag{3}$$

These hypotheses on E define 2D PF considering the minimization of two objectives [3,11]. Such configuration is illustrated by Figure 1. Without loss of generality, transforming objectives to maximize f into $-f$ allows for the consideration of the minimization of two objectives. This assumption impacts the sense of the inequalities of $\mathcal{I}, \prec$. A PF can also be seen as a Skyline operator [12]. A 2D PF can be extracted from any subset of $\mathbb{R}^2$ using an output-sensitive algorithm [13], or using any MOO approach [3,14].

The results of this paper will be given using the Chebyshev and Minkowski distances, generically denoting $d(y,z)$ the l_∞ and l_m norm-induced distance, respectively. For a given $m > 0$, the Minkowski distance is denoted d_m, and given by the formula

$$\forall y = (y^1, y^2), z = (z^1, z^2) \in \mathbb{R}^2, \quad d_m(y,z) = \sqrt[m]{|y^1 - z^1|^m + |y^2 - z^2|^m} \tag{4}$$

The case $m = 2$ corresponds to the Euclidean distance; it is a usual case for our application. The limit with $m \to \infty$ defines the Chebyshev distance, denoted d_∞ and given by the formula

$$\forall y = (y^1, y^2), z = (z^1, z^2) \in \mathbb{R}^2, \quad d_\infty(y, z) = \max\left(\left|y^1 - z^1\right|, \left|y^2 - z^2\right|\right) \tag{5}$$

Once a distance d is defined, a dissimilarity among a subset of points $E' \subset E$ is defined using the radius of the minimal enclosing ball containing E'. Numerically, this dissimilarity function, denoted as $f_{\mathcal{C}}$, can be written as

$$\forall E' \subset E, \quad f_{\mathcal{C}}(E') = \min_{y \in \mathbb{R}^2} \max_{x \in E'} d(x, y) \tag{6}$$

A discrete variant considers enclosing balls with one of the given points as the center. Numerically, this dissimilarity function, denoted $f_{\mathcal{D}}$, can be written as

$$\forall E' \subset E, \quad f_{\mathcal{D}}(E') = \min_{y \in E'} \max_{x \in E'} d(x, y) \tag{7}$$

For the sake of having unified notations for common results and proofs, we define $\gamma \in \{0, 1\}$ to indicate which version of the dissimilarity function is considered. $\gamma = 0$ (respectively, 1) indicates that the continuous (respectively, discrete) version is used, f_γ, thus denoting $f_1 = f_{\mathcal{C}}$ (respectively, $f_0 = f_{\mathcal{D}}$). Note that $\gamma \in \{0, 1\}$ will be related to complexity results which motivated such a notation choice.

For each a subset of points $E' \subset E$ and integer $K \geqslant 1$, we define $\Pi_K(E)$, as the set of all the possible partitions of E' in K subsets. Continuous and discrete K-center are optimization problems with $\Pi_K(E)$ as a set of feasible solutions, covering E with K identical balls while minimizing the radius of the balls used

$$\min_{\pi \in \Pi_K(E)} \max_{P \in \pi} f_\gamma(P) \tag{8}$$

The continuous and discrete K-center problems in the 2D PF are denoted K-γ-CP2DPF. Another covering variant, denoted min-sum-K-radii problems, covers the points with non-identical balls, while minimizing the sum of the radius of the balls. We consider the following extension of min-sum-K-radii problems, with $\alpha > 0$ being a real number

$$\min_{\pi \in \Pi_K(E)} \sum_{P \in \pi} f_\gamma(P)^\alpha \tag{9}$$

$\alpha = 1$ corresponds to the standard min-sum-K-radii problem. $\alpha = 2$ with the standard Euclidean distance is equivalent to the minimization of the area defined by the covering disks. For the sake of unifying notations for results and proofs, we define a generic operator $\oplus \in \{+, \max\}$ to denote, respectively, sum-clustering and max-clustering. This defines the generic optimization problems

$$\min_{\pi \in \Pi_K(E)} \bigoplus_{P \in \pi} f_\gamma(P)^\alpha \tag{10}$$

Lastly, we consider a partial clustering extension of problems (10), similarly to the partial p-center [15]. The covering with balls mainly concerns the extreme points, which make the results highly dependent on outliers. One may consider that a certain number $M < N$ of the points may be considered outliers, and that M points can be removed in the evaluation. This can be written as

$$\min_{E' \subset E : |E \setminus E'| \leqslant M} \; \min_{\pi \in \Pi_K(E')} \bigoplus_{P \in \pi} f_\gamma(P)^\alpha \tag{11}$$

Problem (11) is denoted K-M-$\oplus$-(α,γ)-BC2DPF. Sometimes, the partial covering is defined by a maximal percentage of outliers. In this case, if M is much smaller than N, we have $M = \Theta(N)$, which we have to keep in mind for the complexity results. K-center problems, K-γ-CP2DPF, are K-M-max-(α,γ)-BC2DPF problems for all $\alpha > 0$; the value of α does not matter for max-clustering, defining the same optimal solutions as $\alpha = 1$. The standard min-sum-k-radii problem, equivalent to the min-sum diameter problem, corresponds to k-0-$+$-$(1,\gamma)$-BC2DPF problems for discrete and continuous versions, k-M-$+$-$(1,\gamma)$-BC2DPF problems consider partial covering for min-sum-k-radii problems.

3. Related Works

This section describes works related to our contributions, presenting the state-of-the-art for p-center problems and clustering points in a PF. For more detailed survey on the results for the p-center problems, we refer to [16].

3.1. Solving P-Center Problems and Complexity Results

Generally, the p-center problem consists of locating p facilities among possible locations and assigning n clients, called $c_1, c_2, \ldots, c_n$, to the facilities in order to minimize the maximum distance between a client and the facility to which it is allocated. The continuous p-center problem assumes that any place of location can be chosen, whereas the discrete p-center problem considers a subset o m potential sites, denoted $f_1, f_2, \ldots, f_m$, and distances $d_{i,j}$ for all $i \in [\![1,n]\!]$ and $j \in [\![1,m]\!]$. Discrete p-center problems can be formulated with bipartite graphs, modeling that si unfeasibile for some assignments. In the discrete p-center problem defined in Section 2, points $f_1, f_2, \ldots, f_m$ are exactly $c_1, c_2, \ldots, c_n$, and the distances are defined using a norm, so that triangle inequality holds for such variants.

P-center problems are NP-hard [6,17]. Furthermore, for all $\alpha < 2$, any α-approximation for the discrete p-center problem with triangle inequality is NP-hard [18]. Two approximations were provided for the discrete p-center problem running in $O(pn\log n)$ time and in $O(np)$ time, respectiveltly [19,20]. The discrete p-center problem in $\mathbb{R}^2$ with a Euclidean distance is also NP-hard [17]. Defining binary variables $x_{i,j} \in \{0,1\}$ and $y_j \in \{0,1\}$ with $x_{i,j} = 1$ if and only if the customer i is assigned to the depot j, and $y_j = 1$ if and only if location f_j is chosen as a depot, the following Integer Linear Programming (ILP) formulation models the discrete p-center problem [21]

$$
\begin{aligned}
\min_{x,y,z} \quad & z && && \text{(12a)}\\
s.t: \quad \sum_{j=1}^{n} d_{i,j}x_{i,j} & \leqslant z && \forall i \in [\![1,n]\!] && \text{(12b)}\\
\sum_{j=1}^{m} y_j & = p && && \text{(12c)}\\
\sum_{j=1}^{m} x_{i,j} & = 1 && \forall i \in [\![1,n]\!] && \text{(12d)}\\
x_{i,j} & \leqslant y_j && \forall (i,j) \in [\![1,N]\!] \times [\![1,n]\!], && \text{(12e)}\\
x_{i,j}, y_j & \in \{0,1\} && \forall i,j \in [\![1,n]\!] \times [\![1,m]\!], && \text{(12f)}
\end{aligned}
$$

Constraints (12b) are implied by a standard linearization of the min–max original objective function. Constraint (12c) fixes the number of open facilities to p. Constraints (12d) assign each client to exactly one facility. Constraints (12e) are necessary to induce that any considered assignment $x_{i,j} = 1$ implies that facility j is open with $y_j = 1$. Tighter ILP formulations than (12) were proposed, with efficient exact algorithms relying on the IP models [22,23]. Exponential exact algorithms were also designed for the continuous p-center problem [24,25]. An $n^{O(\sqrt{p})}$-time algorithm was provided for the continuous Euclidean p-center problem in the plane [26]. An $n^{O(p^{1-1/d})}$-time algorithm is available for the continuous p-center problem in $\mathbb{R}^d$ under Euclidean and L_∞ -metric [27].

Specific cases of p-center problems are solvable in polynomial time. The continuous 1-center problem is exactly the minimum covering ball problem, which has a linear complexity in $\mathbb{R}^2$. Indeed, a "prune and search" algorithm finds the optimum bounding sphere and runs in linear time if the dimension is fixed as a constant [28]. In dimension d, its complexity is in $O((d+1)(d+1)!n)$ time, which is impractical for high-dimensional applications [28]. The discrete 1-center problem is solved in time $O(n \log n)$, using furthest-neighbor Voronoi diagrams [29]. The continuous and planar 2-center problem is solved in randomized expected $O(n \log^2 n)$ time [30,31]. The discrete and planar 2-center problem is solvable in $O(n^{4/3} \log^5 n)$ time [32].

1D p-center problems, or those with equivalent points that are located in a line, have specific complexity results with polynomial DP algorithms. The discrete 1D k-center problem is solvable in $O(n)$ time [33]. The continuous and planar k-centers on a line, finding k disks with centers on a given line l, are solvable in polynomial time, in $O(n^2 \log^2 n)$ time in the first algorithm by [29], and in $O(nk \log n)$ time and $O(n)$ space in the improved version provided by [34]. An intensively studied extension of the 1D sub-cases is the p-center in a tree structure. The continuous p-center problem is solvable in $O(n \log^3 n)$ time in a tree structure [7]. The discrete p-center problem is solvable in $O(n \log n)$ time in a tree structure [35].

Rectilinear p-center problems, using the Chebyshev distances, were less studied. Such distance is useful for complexity results; however, it has fewer applications than Euclidean or Minkowski norms. For the planar and rectangular 1-center and 2-center problems, $O(n)$ algorithms are available for the 1-center problem, and such 3-center problems can be solved in $O(n \log n)$ time [36]. In a general dimension d, continuous and discrete versions of rectangular p-center problems are solvable in $O(n)$ and $O(n \log^{d-2} n \log\log n + n \log n)$ running time, respectively. Specific complexity results for rectangular 2-center problems are also available [37].

3.2. *Solving Variants of P-Center Problems and Complexity Results*

Variants of p-center problems were studied less intensively than the standard p-center problems. The partial variants were introduced in 1999 by [15], whereas a preliminary work in 1981 considered a partial weighted one-center variant and a DP algorithm to solve it running in $O(n^2 \log n)$ time [38]. The partial discrete p-center can formulated as an ILP starting from the formulation provided by [21] as written in (12). Indeed, considering that n_0 points can be uncovered, constraints (12.4) become inequalities $\sum_{j=1}^{m} x_{i,j} \leqslant 1$ for all i, j and the maximal number of unassigned points is set to n_0, adding one constraint $\sum_{j=i}^{n} \sum_{j=1}^{m} x_{i,j} \geqslant n - n_0$. Similarly, the sum-clustering variants K-M-+-(α,γ)-BC2DPF can be written as the following ILP

$$
\begin{array}{llllr}
\min\limits_{z,r \geqslant 0} & \displaystyle\sum_{n=1}^{N} r_n & & & \text{(13a)} \\
s.t: & \displaystyle\sum_{n=1}^{N} d(x_n, x_{n'})^{\alpha} z_{n,n'} & \leqslant r_{n'} & \forall n' \in [\![1,N]\!] & \text{(13b)} \\
 & \displaystyle\sum_{n'=1}^{N} z_{n',n'} & = K & & \text{(13c)} \\
 & \displaystyle\sum_{n'=1}^{N} z_{n,n'} & \leqslant 1 & \forall n \in [\![1,N]\!] & \text{(13d)} \\
 & \displaystyle\sum_{n=1}^{N} \sum_{n'=1}^{N} z_{n,n'} & \geqslant N - M & & \text{(13e)} \\
 & z_{n,n'} & \leqslant z_{n',n'} & \forall (n,n') \in [\![1,N]\!]^2, & \text{(13f)} \\
 & z_{n,n'} & \in \{0,1\} & \forall (n,n') \in [\![1,N]\!]^2, & \text{(13g)} \\
 & r_n & \geqslant 0 & \forall n \in [\![1,N]\!], & \text{(13h)}
\end{array}
$$

In this ILP formulation, binary variables $z_{n,n'} \in \{0,1\}$ are defined such that $z_{n,n'} = 1$ if and only if the points x_n and $x_{n'}$ are assigned in the same cluster, with $x_{n'}$ being the discrete center. Continuous variables $r_n \geqslant 0$ denote the powered radius of the ball centered in x_n, if x_n is chosen as a center, and $r_n = 0$ otherwise. Constraint (13b) is a standard linearization of the non-linear objective function. $z_{n,n}$ indicates that if point x_n is chosen as the center, then this implies with (13c) that K such variables are nonzero, and with (13f) that a nonzero variable $z_{n,n'}$ implies that the corresponding $z_{n',n'}$ is not null. (13d) and (13e) allow the extension with partial variants, as discussed before.

Min-sum radii or diameter problems were rarely studied. However, such objective functions are useful for meta-heuristics to break some "plateau" effects [39]. Min-sum diameter clustering is NP-hard in the general case and polynomial within a tree structure [40]. The NP-hardness is also proven, even in metrics induced by weighted planar graphs [41]. Approximation algorithms were studied for min-sum diameter clustering. A logarithmic approximation with a constant factor blowup in the number of clusters was provided by [42]. In the planar case with Euclidean distances, a polynomial time approximation scheme was designed [43].

3.3. *Clustering/Selecting Points in Pareto Frontiers*

Here, we summarize the results related to the selection or the clustering of points in PF, with applications for MOO algorithms. Polynomial complexity resulting in the use of 2D PF structures is an interesting property; clustering problems have a NP-hard complexity in general [17,44,45].

To the best of our knowledge, no specific work focused on PF sub-cases of k-center problems and variants before our preliminary work [10]. A Distance-Based Representative Skyline with similarities to the discrete p-center problem in a 2D PF may not be fully available in the Skyline application, which makes a significant difference [46,47]. The preliminary results proved that K-γ-CP2DPF is solvable in $O(KN \log^{\gamma} N)$ time using $O(N)$ additional memory space [10]. Partial extensions and min-sum-k-radii variants were not considered for 2D PF. We note that the 2D PF case is an extension of the 1D case, with 1D cases being equivalent to the cases of an affine 2D PF. In the study of complexity results, a tree structure is usually a more studied extension of 1D cases. The discrete k-center problem on a tree structure, and thus the 1D sub-case, is solvable in $O(N)$ time [33]. 3F PF cases are NP-complete, as already mentioned in the introduction, this being a consequence of the NP-hardness of the general planar case.

Maximization of the quality of discrete representations of Pareto sets was studied with the hypervolume measure in the Hypervolume Subset Selection (HSS) problem [48,49]. The HSS problem is known to be NP-hard in dimension 3 (and greater dimensions) [50]. HSS is solvable with an exact algorithm in $N^{O(\sqrt{K})}$ and a polynomial-time approximation scheme for any constant dimension d [50]. The 2D case is solvable in polynomial time with a DP algorithm with a complexity in $O(KN^2)$ time and $O(KN)$ space [49]. The time complexity of the DP algorithm was improved in $O(KN + N \log N)$ by [51], and in $O(K(N-K) + N \log N)$ by [52].

The selection of points in a 2D PF, maximizing the diversity, can also be formulated using p-dispersion problems. Max–Min and Max-Sum p-dispersion problems are NP-hard problems [53,54]. Max–Min and Max-Sum p-dispersion problems are still NP-hard problems when distances fulfill the triangle inequality [53,54]. The planar (2D) Max–Min p-dispersion problem is also NP-hard [9]. The one-dimensional (1D) cases of Max–Min and Max-Sum p-dispersion problems are solvable in polynomial time, with a similar DP algorithm running in $O(\max\{pN, N \log N\})$ time [8,9]. Max–Min p-dispersion was proven to be solvable in polynomial time, with a DP algorithm running in $O(pN \log N)$ time and $O(N)$ space [55]. Other variants of p-dispersion problems were also proven to be solvable in polynomial time using DP algorithms [55].

Similar results exist for k-means, k-medoid and k-median clustering. K-means is NP-hard for 2D cases, and thus for 3D PF [44]. K-median and K-medoid problems are known to be NP hard in dimension 2, since [17], where the specific case of 2D PF was proven to be solvable in $O(N^3)$ time with DP algorithms [11,56]. The restriction of k-means to 2D PF would be also solvable in $O(N^3)$ time with a DP algorithm if a conjecture was proven [57]. We note that an affine 2D PF is a line in $\mathbb{R}^2$, where clustering is equivalent to 1D cases. 1D k-means were proven to be solvable in polynomial time with a DP algorithm in $O(KN^2)$ time and $O(KN)$ space. This complexity was improved for a DP algorithm in $O(KN)$ time and $O(N)$ space [58]. This is thus the complexity of K-means in an affine 2D PF.

4. Intermediate Results

4.1. Indexation and Distances in a 2D PF

Lemma 1. *$\preccurlyeq$ is an order relation, and $\prec$ is a transitive relation*

$$\forall x,y,z \in \mathbb{R}^2, \quad x \prec y \;\; and \;\; y \prec z \Longrightarrow x \prec z \tag{14}$$

Proposition 1 implies an order among the points of E, for a re-indexation in $O(N \log N)$ time

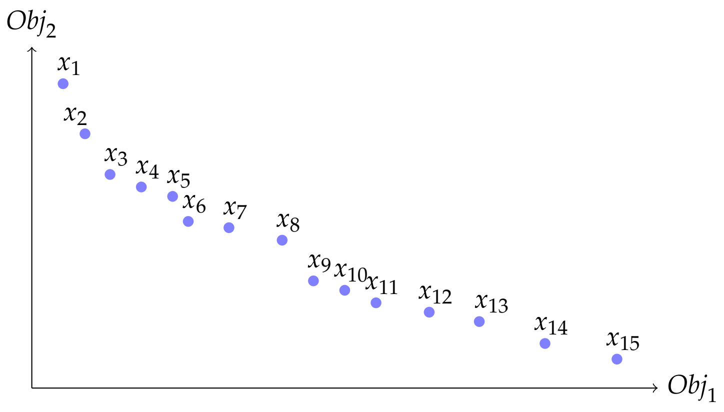

Figure 1. Illustration of a 2D Pareto Front (PF) with 15 points and the indexation implied by Proposition 1.

Proposition 1 (Total order). *Points (x_i) can be re-indexed in $O(N \log N)$ time, such that*

$$\forall (i_1,i_2) \in [\![1;N]\!]^2, \quad i_1 < i_2 \Longrightarrow \quad x_{i_1} \prec x_{i_2} \tag{15}$$
$$\forall (i_1,i_2) \in [\![1;N]\!]^2, \quad i_1 \leqslant i_2 \Longrightarrow \quad x_{i_1} \preccurlyeq x_{i_2} \tag{16}$$

Proof. We index E such that the first coordinate is increasing. This sorting procedure runs in $O(N \log N)$ time. Let $(i_1,i_2) \in [\![1;N]\!]^2$, with $i_1 < i_2$. We thus have $x^1_{i_1} < x^1_{i_2}$. Having $x_{i_1} \mathcal{I} x_{i_2}$ implies that $x^2_{i_1} > x^2_{i_2}$. $x^1_{i_1} < x^1_{i_2}$ and $x^2_{i_1} > x^2_{i_2}$ is by definition $x_{i_1} \prec x_{i_2}$. $\square$

The re-indexation also implies monotonic relations among distances of the 2D PF

Lemma 2. *We suppose that E is re-indexed as in Proposition 1. Letting d be a Minkowski, Euclidean or Chebyshev distance, we obtain the following monotonicity relations*

$$\forall (i_1,i_2,i_3) \in [\![1;N]\!]^3, \quad i_1 \leqslant i_2 < i_3 \Longrightarrow d(x_{i_1},x_{i_2}) < d(x_{i_1},x_{i_3}) \tag{17}$$
$$\forall (i_1,i_2,i_3) \in [\![1;N]\!]^3, \quad i_1 < i_2 \leqslant i_3 \Longrightarrow d(x_{i_2},x_{i_3}) < d(x_{i_1},x_{i_3}) \tag{18}$$

Proof. We first note that the equality cases are trivial, so we can suppose that $i_1 < i_2 < i_3$ in the following proof. We prove the propriety (17); the proof of (18) is analogous.

Let $i_1 < i_2 < i_3$. We note that $x_{i_1} = (x^1_{i_1}, x^2_{i_1})$, $x_{i_2} = (x^1_{i_2}, x^2_{i_2})$ and $x_{i_3} = (x^1_{i_3}, x^2_{i_3})$. Proposition 1 re-indexation ensures $x^1_{i_1} < x^1_{i_2} < x^1_{i_3}$ and $x^2_{i_1} > x^2_{i_2} > x^2_{i_3}$. With $x^1_{i_3} - x^1_{i_1} > x^1_{i_2} - x^1_{i_1} > 0$, $|x^1_{i_1} - x^1_{i_2}| < |x^1_{i_1} - x^1_{i_3}|$

With $x^2_{i_3} - x^2_{i_1} < x^2_{i_2} - x^2_{i_1} < 0$, $|x^2_{i_1} - x^2_{i_2}| < |x^2_{i_1} - x^2_{i_3}|$

Thus, for any $m > 0$, $d_m(x_{i_1}, x_{i_2}) < |x^1_{i_1} - x^1_{i_3}|^m + |x^2_{i_1} - x^2_{i_3}|^m = d_m(x_{i_1}, x_{i_3})$ and also

$d_\infty(x_{i_1}, x_{i_2}) = \max(|x^1_{i_1} - x^1_{i_2}|, |x^2_{i_1} - x^2_{i_2}|) < \max(|x^1_{i_1} - x^1_{i_3}|, |x^2_{i_1} - x^2_{i_3}|) = d_\infty(x_{i_1}, x_{i_3})$. Hence, the result is proven for Euclidean, Minkowski and Chebyshev distances. □

4.2. Lemmas Related to Cluster Costs

This section provides the relations needed to compute or compare cluster costs. Firstly, one notes that the computation of cluster costs is easy in a 2D PF in the continuous clustering case.

Lemma 3. *Let $P \subset E$, such that $card(P) \geqslant 1$. Let i (resp i') be the minimal (respective maximal) index of points of P with the indexation of Proposition 1. Then, $f_{\mathcal{C}}(P)$ can be computed with $f_{\mathcal{C}}(P) = \frac{1}{2} d(x_i, x_{i'})$.*

To prove the Lemma 3, we use the Lemmas 4 and 5.

Lemma 4. *Let $P \subset E$, such that $card(P) \geqslant 1$. Let i (resp i') the minimal (resp maximal) index of points of P with the indexation of Proposition 1. We denote with $O = \frac{x_i + x_{i'}}{2}$ the midpoint of $x_i, x_{i'}$. Then, using a Minkowski or Chebyshev distance d, we have for all $x \in P$: $d(x, O) \leqslant d(x_i, O) = d(x_{i'}, O)$.*

Proof of Lemma 4: We denote with $r = d(x_i, O) = d(x_{i'}, O) = \frac{1}{2} d(x_i, x_{i'})$, with the equality being trivial as points $O, x_i, x_{i'}$ are on a line and d is a distance. Let $x \in P$. We calculate the distances using a new system of coordinates, translating the original coordinates such that O, is a new origin (which is compatible with the definition of Pareto optimality). x_i and $x_{i'}$ have coordinates $(-a, b)$ and $(a, -b)$ in the new coordinate system, with $a, b > 0$ and $a^m + b^m = r^m$ if a Minkowski distance is used, otherwise it is $\max(a, b) = r$ for the Chebyshev distance. We use (a', b') to denote the coordinates of x. $x_i \prec x \prec x_{i'}$ implies that $-a \leqslant a' \leqslant a$ and $-b \leqslant b' \leqslant b$, i.e., $|a'| \leqslant a$ and $|b'| \leqslant b$, which implies $d(x, O) \leqslant r$, using Minkowski or Chebyshev distances. □

Lemma 5. *Let $P \subset E$ such that $card(P) \geqslant 1$. Let i (respective i') be the minimal (respective maximal) index of points of P with the indexation of Proposition 1. We denote, using $O = \frac{x_i + x_{i'}}{2}$, the midpoint of $x_i, x_{i'}$. Then, using a Minkowski or Chebyshev distance d, we have for all $y \in \mathbb{R}^2$: $d(x_i, O) = d(x_{i'}, O) \leqslant \max(d(x_i, y), d(x_{i'}, y))$.*

Proof of Lemma 5: As previously noted, let $r = d(x_i, O) = d(x_{i'}, O) = \frac{1}{2} d(x_i, x_{i'})$. Let $y \in \mathbb{R}^2$. We have to prove that $d(x_{i'}, y) \geqslant r$ or $d(x_i, y) \geqslant r$. If we suppose that $d(x_i, y) < r$, this implies that $y \prec O$. Then, having $y \prec O \prec x_{i'}$ implies $d(x_{i'}, y) > d(x_{i'}, 0) = r$ with Lemma 2. □

Proof of Lemma 3: We first note that $f_{\mathcal{C}}(P) = \min_{y \in \mathbb{R}^2} \max_{x \in P} d(x, y) \leqslant \max_{x \in P} d(x, O)$, using the particular point $O = \frac{x_i + x_{i'}}{2}$. Using Lemma 4, $\max_{x \in P} d(x, O) \leqslant r$, and thus $f_{\mathcal{C}}(P) \leqslant r$ with $r = d(x_i, O) = d(x_{i'}, O) = \frac{1}{2} d(x_i, x_{i'})$. Reciprocally, for all $y \in \mathbb{R}^2$, $r \leqslant \max(d(x_i, y), d(x_{i'}, y))$ using Lemma 5, and thus $r \leqslant \max_{x \in P} d(x, y)$. This implies that $r \leqslant \min_{y \in \mathbb{R}^2} \max_{x \in P} d(x, y) = f_{\mathcal{C}}(P)$. □

Lemma 6. *Let $P \subset E$ such that $card(P) \geqslant 3$. Let i (respective i′) the minimal (respective maximal) index of points of P.*

$$f_{\mathcal{D}}(P) = \min_{j \in [\![i+1, i'-1]\!], x_j \in P} \max\big(d(x_j, x_i), d(x_j, x_{i'})\big) \tag{19}$$

Proof. Let $y \in P - \{x_i, x_{i'}\}$. We denote $j \in [\![i, i']\!]$, such that $y = x_j$. Applying Lemma 2 to $i < j < i'$, for all $k \in [\![i, i']\!]$,, we have $d(x_j, x_k) \leqslant \max\big(d(x_j, x_i), d(x_j, x_i)\big)$. Then

$$\begin{aligned} f_{\mathcal{D}}(P) &= \min_{y = x_j \in P} \max_{x \in P} d(x, y) \\ f_{\mathcal{D}}(P) &= \min_{j \in [\![i, i']\!], x_j \in P} \max\left(\max\big(d(x_j, x_i), d(x_j, x_i)\big), \max_{k \in [\![i, i']\!]} d(x_j, x_k)\right) \\ f_{\mathcal{D}}(P) &= \min_{j \in [\![i, i']\!], x_j \in P} \max\big(d(x_j, x_i), d(x_j, x_{i'})\big) \end{aligned}$$

Lastly, we notice that extreme points are not optimal centers. Indeed, $\max(d(x_i, x_i), d(x_i, x_{i'})) = d(x_i, x_{i'}) > \max(d(x_{i+1}, x_{i'}), d(x_{i+1}, x_i))$ with Proposition 2, i.e., i, is not optimal in the last minimization, dominated by $i+1$. Similarly, i' is dominated by $i' - 1$. □

Lemma 7. *Let $\gamma \in \{0, 1\}$. Let $P \subset P' \subset E$. We have $f_\gamma(P) \leqslant f_\gamma(P')$.*

Proof. Using the order of Proposition 1, let i (respectively, i') the minimal index of points of P (respectively, P') and let j (respectively, j') the maximal indexes of points of P (respectively, P'). $f_{\mathcal{C}}(P) \leqslant f_{\mathcal{C}}(P')$ is trivial using Lemmas 2 and 3. To prove $f_{\mathcal{D}}(P) \leqslant f_{\mathcal{D}}(P')$, we use $i \leqslant i' \leqslant j' \leqslant j$, and Lemmas 2 and 6

$$\begin{aligned} f_{\mathcal{D}}(P) &= \min_{k \in [\![i, j]\!], x_k \in P} \max\big(d(x_k, x_i), d(x_j, x_k)\big) \\ &\leqslant \min_{k \in [\![i', j']\!], x_k \in P} \max\big(d(x_k, x_i), d(x_j, x_k)\big) \\ &\leqslant \min_{k \in [\![i', j']\!], x_k \in P'} \max\Big(d(x_k, x_{i'}), d\Big(x_{j'}, x_k\Big)\Big) = f_{\mathcal{D}}(P') \end{aligned}$$

□

Lemma 8. *Let $\gamma \in \{0, 1\}$. Let $P \subset E$, such that $card(P) \geqslant 1$. Let i (respectively, i′) the minimal (respectively, maximal) index of points of P. For all $P' \subset P$, such that $x_i, x_{i'} \in P'$, we have $f_\gamma(P) = f_\gamma(P')$*

Proof. Let $P' \subset P$ such that $x_i, x_{i'} \in P'$. With Lemma 7, we have $f_\gamma(P') \leqslant f_\gamma(P)$. $f_{\mathcal{C}}(P') = f_{\mathcal{C}}(P)$ is trivial using Lemma 3, so that we have to prove $f_{\mathcal{D}}(P) \leqslant f_{\mathcal{D}}(P')$.
$f_{\mathcal{D}}(P) = \min_{k \in [\![i, i']\!], x_k \in P} \max(d(x_k, x_i), d(x_i - x_{i'})) \leqslant \min_{k \in [\![i', j']\!], x_k \in P'} \max(d(x_k, x_i), d(x_k, x_{i'})) = f_{\mathcal{D}}(P')$ □

4.3. Optimality of Non-Nested Clustering

In this section, we prove that non-nested clustering property, the extension of interval clustering from 1D to 2D PF, allows the computation of optimal solutions, which will be a key element for a DP algorithm. For (partial) p-center problems, i.e., K-M-max-(α, γ)-BC2DPF, optimal solutions may exist without fulfilling the non-nested property, whereas for K-M-$+$-$(\alpha, 0)$-BC2DPF problems, the nested property is a necessary condition to obtaining an optimal solution.

Lemma 9. *Let $\gamma \in \{0,1\}$; let $M > 0$. There is an optimal solution of* 1-*M*-$\oplus$-(α,γ)*-BC2DPF on the shape $\mathcal{C}_{i,i'} = \{x_j\}_{j \in [\![i,i']\!]} = \{x \in E \mid \exists j \in [\![i,i']\!],\, x = x_j\}$, with $|i'-i| \geqslant N - M$.*

Proof. Let $\pi \in \Pi_K(E)$ represent an optimal solution of 1-M-$\oplus$-(α,γ)-BC2DPF, let OPT be the optimal cost, and $\mathcal{C} \subset E$ with $|\mathcal{C}| \geqslant N - M$ and $f_\gamma(\mathcal{C}) = OPT$. Let i (respectively, i') be the minimal (respectively maximal) index of $\mathcal{C}$ using order of Proposition 1. $\mathcal{C} \subset \mathcal{C}_{i,i'}$, so Lemma 8 applies and $f_\gamma(\mathcal{C}_{i,i'}) = f_\gamma(\mathcal{C}) = OPT$. $|\mathcal{C}_{i,i'}| \geqslant |\mathcal{C}| \geqslant N - M$, thus $\mathcal{C}_{i,i'}$ defines an optimal solution of 1-M-$\oplus$-(α,γ)-BC2DPF. □

Proposition 2. *Let $E = (x_i)$ be a 2D PF, re-indexed with Proposition 1. There are optimal solutions of K-M-$\oplus$-(α,γ)-BC2DPF using only clusters on the shape $\mathcal{C}_{i,i'} = \{x_j\}_{j \in [\![i,i']\!]} = \{x \in E \mid \exists j \in [\![i,i']\!],\, x = x_j\}$.*

Proof. We prove the results by the induction on $K \in \mathbb{N}^*$. For $K = 1$, Lemma 9 gives the initialization.

Let us suppose that $K > 1$ and the Induction Hypothesis (IH) that Proposition 2 is true for K-M-$\oplus$-(α,γ)-BC2DPF. Let $\pi \in \Pi_K(E)$ be an optimal solution of K-M-$\oplus$-(α,γ)-BC2DPF; let OPT be the optimal cost. Let $X \subset E$ be the subset of the non-selected points, $|X| \leqslant M$, and $\mathcal{C}_1, \ldots, \mathcal{C}_K$ be the K subsets defining the costs, so that $X, \mathcal{C}_1, \ldots, \mathcal{C}_K$ is a partition of E and $\bigoplus_{k=1}^K f_\gamma(\mathcal{C}_k)^\alpha = OPT$. Let N' be the maximal index, such that $x_{N'} \notin X$, which is, necessarily, $N' \geqslant N - M$. We reindex the clusters $\mathcal{C}_k$, such that $x_{N'} \in \mathcal{C}_K$. Let i be the minimal index such that $x_i \in \mathcal{C}_K$.

We consider the subsets $\mathcal{C}'_K = \{x_j\}_{j \in [\![i,N']\!]}$, $X' = X \cap [\![1, i-1]\!]$ and $\mathcal{C}'_k = \mathcal{C}_k \cap \{x_j\}_{j \in [\![1,i-1]\!]}$ for all $k \in [\![1, K-1]\!]$. It is clear that $X', \mathcal{C}'_1, \ldots, \mathcal{C}'_{K-1}$ is a partition of $\{x_j\}_{j \in [\![1,i-1]\!]}$, and $X', \mathcal{C}'_1, \ldots, \mathcal{C}'_K$ is a partition of E. For all $k \in [\![1, K-1]\!]$, $\mathcal{C}'_k \subset \mathcal{C}_k$, so that $f_\gamma(\mathcal{C}'_k) \leqslant f_\gamma(\mathcal{C}_k)$ (Lemma 7).

$X', \mathcal{C}'_1, \ldots, \mathcal{C}'_K$ is a partition of E, and $\bigoplus_{k=1}^K f_\gamma(\mathcal{C}'_k)^\alpha \leqslant OPT$. $\mathcal{C}'_1, \ldots, \mathcal{C}'_K$ is an optimal solution of K-$|X'|$-$\oplus$-(α,γ)-BC2DPF. $\mathcal{C}'_1, \ldots, \mathcal{C}'_{K-1}$ is an optimal solution of $(K-1)$-$|X'|$-$\oplus$-(α,γ)-BC2DPF, applied to points $E' = \cup_{k=1}^{K-1} \mathcal{C}'_1 \cup X'$. Letting OPT' be the optimal cost of $(K-1)$-$|X'|$-$\oplus$-(α,γ)-BC2DPF, we have $OPT = OPT' \oplus f_\gamma(\mathcal{C}'_K)^\alpha$. Applying IH for of $(K-1)$-$|X'|$-$\oplus$-(α,γ)-BC2DPF to points E', we have $\mathcal{C}''_1, \ldots, \mathcal{C}''_{K-1}$ an optimal solution of $(K-1)$-$|X'|$-$\oplus$-(α,γ)-BC2DPF among E' on the shape $\mathcal{C}_{i,i'} = \{x_j\}_{j \in [\![i,i']\!]} = \{x \in E' \mid \exists j \in [\![i,i']\!],\, x = x_j\}$. $\bigoplus_{k=1}^K f_\gamma(\mathcal{C}''_k)^\alpha = OPT'$, and thus $\bigoplus_{k=1}^K f_\gamma(\mathcal{C}''_k)^\alpha \oplus f_\gamma(\mathcal{C}'_K)^\alpha = OPT$. $\mathcal{C}''_1, \ldots, \mathcal{C}''_{K-1}, \mathcal{C}'_K$ is an optimal solution of K-M-$\oplus$-(α,γ)-BC2DPF in E using only clusters $\mathcal{C}_{i,i'}$. Hence, the result is proven by induction. □

Proposition 3. *There is an optimal solution of K-M-$\oplus$-$(\alpha,0)$-BC2DPF, removing exactly M points in the partial clustering.*

Proof. Starting with an optimal solution of K-M-$+$-$(\alpha,0)$-BC2DPF, let OPT be the optimal cost, and let $X \subset E$ be the subset of the non-selected points, $|X| \leqslant M$, and $\mathcal{C}_1, \ldots, \mathcal{C}_K$, the K subsets defining the costs, so that $X, \mathcal{C}_1, \ldots, \mathcal{C}_K$ is a partition of E. Removing random $M - |X|$ points in $\mathcal{C}_1, \ldots, \mathcal{C}_K$, we have clusters $\mathcal{C}'_1, \ldots, \mathcal{C}'_K$ such that, for all $k \in [\![1, K-1]\!]$, $\mathcal{C}'_k \subset \mathcal{C}_k$, and thus $f_\gamma(\mathcal{C}'_k) \leqslant f_\gamma(\mathcal{C}_k)$ (Lemma 7). This implies $\bigoplus_{k=1}^K f_\gamma(\mathcal{C}'_k)^\alpha \leqslant \bigoplus_{k=1}^K f_\gamma(\mathcal{C}_k)^\alpha = OPT$, and thus the clusters $\mathcal{C}'_1, \ldots, \mathcal{C}'_K$ and outliers $X' = E \setminus \cup_k \mathcal{C}'_k$ define and provide the optimal solution of K-M-$\oplus$-$(\alpha,0)$-BC2DPF with exactly M outliers. □

Reciprocally, one may investigate if the conditions of optimality in Propositions 2 and 3 are necessary. The conditions are not necessary in general. For instance, with $E = \{(3,1); (2,2); (1,3)\}$, $K = M = 1$ and the discrete function $F_{\mathcal{D}}$, ie $\gamma = 1$, the selection of each pair of points defines an optimal solution, with the same cost as the selection of the three points, which do not fulfill the property of Proposition 3. Having an optimal solution with the two extreme points also does not fulfill the property of Proposition 2. The optimality conditions are necessary in the case of sum-clustering, using the continuous measure of the enclosing disk.

Proposition 4. *Let an optimal solution of* K*-*M*-+-*$(\alpha,0)$*-BC2DPF be defined with* $X \subset E$ *as the subset of outliers, with* $|X| \leqslant M$*, and* $\mathcal{C}_1, \dots, \mathcal{C}_K$ *as the* K *subsets defining the optimal cost. We therefore have*

(i) $\left|\bigcup_{k=1}^{K}\mathcal{C}_k\right| = M$*, in other words, exactly* M *points are not selected in* π*.*

(ii) *For each* $k \in [\![1,K]\!]$*, defining* $i_k = \min\{i \in [\![1,N]\!] | x_i \in \mathcal{C}_k\}$ *and* $j_k = \max\{i \in [\![1,N]\!] | x_i \in \mathcal{C}_k\}$*, we have* $\mathcal{C}_k = \{x_i\}_{i \in [\![i_k,j_k]\!]}$*.*

Proof. Starting with an optimal solution of K-M-+-$(\alpha,0)$-BC2DPF, let OPT be the optimal cost, and let $X \subset E$ be the subset of the non-selected points, $|X| \leqslant M$, and $\mathcal{C}_1, \dots, \mathcal{C}_K$ be the K subsets defining the costs, so that $X, \mathcal{C}_1, \dots, \mathcal{C}_K$ is a partition of E. We prove (i) and (ii) ad absurdum.

If $|X| < M$, one may remove one extreme point of the cluster $\mathcal{C}_1$, defining $\mathcal{C}'_1$. With Lemmas 2 and 3, we have $f_{\mathcal{C}}(\mathcal{C}'_1) < f_{\mathcal{C}}(\mathcal{C}_1)$, and $f_{\mathcal{C}}(\mathcal{C}'_1)^\alpha + \sum_{k=1}^{K} f_{\mathcal{C}}(\mathcal{C}_k)^\alpha < f_{\mathcal{C}}(\mathcal{C}_1)^\alpha + \sum_{k=1}^{K} f_{\mathcal{C}}(\mathcal{C}_k)^\alpha = OPT$. This is in contraction with the optimality of $\mathcal{C}_1, \dots, \mathcal{C}_K, \mathcal{C}'_1, \mathcal{C}_2 \dots, \mathcal{C}_K$, defining a strictly better solution for K-M-+-$(\alpha,0)$-BC2DPF. (i) is thus proven ad absurdum.

If (ii) is not fulfilled by a cluster $\mathcal{C}_k$, there is $x_i \notin \mathcal{C}_k$ with $i \in [\![i_k, j_k]\!]$. If $x_i \in X$, we have a better solution than the optimal one with $X' = X \cup \{x_{i_k}\} \setminus \{x_i\}$ and $\mathcal{C}'_k = \mathcal{C}_k \cup \{x_i\} \setminus \{x_{i_k}\}$. If $x_i \in \mathcal{C}_l$ with $l \neq k$, we have nested clusters $\mathcal{C}_l$ and $\mathcal{C}_k$. We suppose that $i_k < i_l$ (otherwise, reasoning is symmetrical). We define a better solution than the optimal one with $\mathcal{C}'_l = \mathcal{C}_k \cup \{x_i\} \setminus \{x_{i_l}\}$ and $\mathcal{C}'_k = \mathcal{C}_k \cup \{x_{i_l}\} \setminus \{x_i\}$. (ii) is thus proven ad absurdum. $\square$

4.4. Computation of Cluster Costs

Using Proposition 2, only cluster costs $\mathcal{C}_{i,i'}$ are computed. This section allows the efficient computation of such cluster costs. Once points are sorted using Proposition 1, cluster costs $f_{\mathcal{C}}(\mathcal{C}_{i,i'})$ can be computed in $O(1)$ using Lemma 3. This makes a time complexity in $O(N^2)$ to compute all the cluster costs $f_{\mathcal{C}}(\mathcal{C}_{i,i'})$ for $1 \leqslant i \leqslant i' \leqslant N$.

Equation (19) ensures that cluster costs $f_{\mathcal{D}}(\mathcal{C}_{i,i'})$ can be computed in $O(i'-i)$ for all $i < i'$. Actually, Algorithm 1 and Proposition 5 allow for computations in $O(\log(i'-i))$ once points are sorted following Proposition 1, with a dichotomic and logarithmic search.

Lemma 10. *Letting* (i,i') *with* $i < i'$*.* $f_{i,i'} : j \in [\![i,i']\!] \longmapsto \max(d(x_j - x_i), d(x_j - x_{i'}))$ *decreases before reaching a minimum* $f_{i,i'}(l)$*,* $f_{i,i'}(l+1) \geqslant f_{i,i'}(l)$*, and then increases for* $j \in [\![l+1,i']\!]$*.*

Proof : We define $g_{i,i',j}, h_{i,i',j}$ with $g_{i,i'} : j \in [\![i,i']\!] \longmapsto d(x_j - x_i)$ and $h_{i,i'} : j \in [\![i,i']\!] \longmapsto d(x_j - x_{i'})$.

Let $i < i'$. Proposition 2, applied to i and any $j, j+1$ with $j \geqslant i$ and $j < i'$, ensures that g is decreasing. Similarly, Proposition 2, applied to i' and any $j, j+1$, ensures that h is increasing.

Let $A = \{j \in [\![i,i']\!] | \forall m \in [\![i,j]\!] g_{i,i'}(m) < h_{i,i'}(m)\}$.$g_{i,i'}(i) = 0$ and $h_{i,i'}(i) = d(x_{i'} - x_i) > 0$, so that $i \in A$. A is a non-empty and bounded subset of $\mathbb{N}$, so that A has a maximum. We note that $l = \max A$.$h_{i,i'}(i') = 0$ and $g_{i,i'}(i') = d(x_{i'} - x_i) > 0$, so that $i' \notin A$ and $l < i'$.

Let $j \in [\![i,l-1]\!]$. $g_{i,i'}(j) < g_{i,i'}(j+1)$ and $h_{i,i',j}(j+1) < h_{i,i'}(j)$, using the monotony of $g_{i,i'}$ and $h_{i,i'}$.$f_{i,i'}(j+1) = \max(g_{i,i'}(j+1), h_{i,i'}(j+1)) = h_{i,i}(j+1)$ and $f_{i,i'}(j) = \max(g_{i,i'}(j), h_{i,i'}(j)) = h_{i,i}(j)$ as $j, j+1 \in A$. Hence, $f_{i,i'}(j+1) = h_{i,i'}(j+1) < h_{i,i'}(j) = f_{i,i'}(j)$. This proves that $f_{i,i'}$ is decreasing in $[\![i,l]\!]$.

$l+1 \notin A$ and $g_{i,i'}(l+1) > h_{i,i'}(l+1)$ have to be coherent with the fact that $l = \max A$.

Let $j \in [\![l+1,i'-1]\!]$. $j+1 > j \geqslant l+1$, so $g_{i,i'}(j+1) > g_{i,i'}(j) \geqslant g_{i,i'}(l+1) > h_{i,i'}(l+1) \geqslant h_{i,i'}(j) > h_{i,i'}(j+1)$ using the monotony of $g_{i,i'}$ and $h_{i,i'}$.This proves that $f_{i,i'}$ is increasing in $[\![l+1,i']\!]$.

Lastly, the minimum of f can be reached in l or in $l+1$, depending on the sign of $f_{i,i'}(l+1) - f_{i,i'}(l)$. If $f_{i,i'}(l+1) = f_{i,i'}(l)$, there are two minimums $l, l+1$. Otherwise, there is a unique minimum $l_0 \in \{l, l+1\}$, $f_{i,i'}$, which decreases before increasing.

Algorithm 1: Computation of $f_{\mathcal{D}}(\mathcal{C}_{i,i'})$

```
input: indexes $i < i'$, a distance $d$
output: the cost $f_{\mathcal{D}}(\mathcal{C}_{i,i'})$

    define $\underline{i} := i$, $\underline{v} := d(x_i - x_{i'})$, $\overline{i} := i'$, $\overline{v} := d(x_i - x_{i'})$,
    while $\overline{i} - \underline{i} \geqslant 2$
        $i'' := \left\lfloor \frac{\overline{i}+\underline{i}}{2} \right\rfloor$
        if $d(x_i - x_{i''}) < d(x_{i'} - x_{i''})$ then $\underline{i} := i''$ and $\underline{v} := d(x_{i'} - x_{i''})$
            else $\overline{i} := i''$ and $\overline{v} := d(x_i - x_{i''})$
    end while
return $\min(\underline{v}, \overline{v})$
```

Proposition 5. *Let $E = \{x_1, \ldots, x_N\}$ be N points of $\mathbb{R}^2$, such that for all $i < j$, $x_i \prec x_j$. The computing cost $f_{\mathcal{D}}(\mathcal{C}_{i,i'})$ for any cluster $\mathcal{C}_{i,i'}$ has a complexity in $O(\log(i'-i))$ time, using $O(1)$ additional memory space.*

Proof. Let $i < i'$. Let us prove the correctness and complexity of Algorithm 1. Algorithm 1 is a dichotomic and logarithmic search; it iterates $O(\log(i'-i))$ times, with each iteration running in $O(1)$ time. The correctness and complexity of Algorithm 1 is a consequence of Lemma 10 and the loop invariant, which exists as a of minimum of $f_{i,i'}$, $f_{i,i'}(j^*)$ with $\underline{i} \leqslant j^* \leqslant \overline{i}$, also having $\underline{v} = f_{i,i'}(\underline{i})$ and $\overline{v} = f_{i,i'}(\overline{i})$. By construction in Algorithm 1, we have $d(x_i - x_{\underline{i}}) < d(x_{i'} - x_{\underline{i}})$, and thus $f_{i,i'}(\underline{i}) = d(x_{i'} - x_{\underline{i}})$. This implies that $f_{i,i'}(\underline{i}-1) = d(x_{i'} - x_{\underline{i}-1}) > f_{i,i'}(\underline{i})$, and thus $\underline{i} \leqslant j^*$, using Lemma 10. Similarly, we always obtain $d(x_i - x_{\underline{i}}) \geqslant d(x_{i'} - x_{\underline{i}})$, and thus $f_{i,i'}(\overline{i}) = d(x_i - x_{\overline{i}})$, $f_{i,i'}(\overline{i}+1) = d(x_i - x_{\overline{i}+1}) > f_{i,i'}(\overline{i})$, so that $\overline{i} \geqslant j^*$ with Lemma 10. At the convergence of the dichotomic search, $\overline{i} - \underline{i} = 1$ and j^* is $\underline{i}$ or $\overline{i}$; therefore, the optimal value is $f_{\mathcal{D}}(\mathcal{C}_{i,i'}) = f_{i,i'}(j^*) = \min(\underline{v}, \overline{v})$. □

Remark 1. *Algorithm 1 improves the previously proposed binary search algorithm [10]. If it has the same logarithmic complexity, this leads to two times fewer calls of the distance function. Indeed, in the previous version, the dichotomic algorithm is computed at each iteration $f_{i,i'}(i'')$ and $f_{i,i'}(i''+1)$ to determine if i'' is in the increasing or decreasing phase of $f_{i,i'}$. In Algorithm 1, the computations that are provided for each iteration are equivalent to the evaluation of only $f_{i,i'}(i'')$, computing $d(x_i - x_{i''})$ and $d(x_{i'} - x_{i''})$.*

Proposition 5 can compute $f_{\mathcal{D}}(\mathcal{C}_{i,i'})$ for all $i < i'$ in $O(N^2 \log N)$. Now, we prove that the costs $f_{\mathcal{D}}(\mathcal{C}_{i,i'})$ of all $i < i'$ can be computed in $O(N^2)$ time instead of $O(N^2 \log N)$ using $O(N^2)$-independent computations. Two schemes are proposed, computing the lines of the cost matrix in $O(N)$ time, computing $f_{\mathcal{D}}(\mathcal{C}_{j,j'})^\alpha$ for all $j' \in [\![j; N]\!]$ for a given $j \in [\![1; N]\!]$ in Algorithm 2, and computing $f_{\mathcal{D}}(\mathcal{C}_{j',j})^\alpha$ for all $j' \in [\![1; j]\!]$ for a given $j \in [\![1; N]\!]$ in Algorithm 3.

Lemma 11. *Let $i, i' \in [\![1, N]\!]$, with $i+1 < i'$. Let $c \in [\![i+1, i'-1]\!]$, such that $f_{i,i'}(c) = f_{\mathcal{D}}(\mathcal{C}_{i,i'})$.*

(i) If $i' < N$, then there is c', such that $c \leqslant c' \leqslant i'$, with $f_{i,i'+1}(c') = \min_{l \in [\![i+1,i'-1]\!]} f_{i,i'}(l) = f_{\mathcal{D}}(\mathcal{C}_{i,i'+1})$.

(ii) If $i > 1$, then there is c'', such that $i \leqslant c' \leqslant c$, with $f_{i-1,i'}(c') = \min_{l \in [\![i+1,i'-1]\!]} f_{i-1,i'}(l) = f_{\mathcal{D}}(\mathcal{C}_{i-1,i'})$.

Proof. We prove (i); we suppose that $i' < N$ and we prove that, for all $c' < c$ $f_{i,i'+1}(c) \leqslant f_{i,i'+1}(c')$, so that either c is an argmin of the minimization, and the superior minimum to

c. (ii) is similarly proven. Let $c' < c$. $f_{i,i'}(c) = f_{\mathcal{D}}(\mathcal{C}_{i,i'})$, which implies $f_{i,i'}(c) \leqslant f_{i,i'}(c')$ and, with Lemma 10, $f_{i,i'}$ is decreasing in $[\![c', c]\!]$, i.e., $f_{i,i'}(c'') = d(x_{c''}, x_{i'})$ for all $c'' \in [\![c', c]\!]$ We thus have $d(x_{c'}, x_{i'}) \geqslant d(x_{c'}, x_i)$, and, with lemma 2, $d(x_{c'}, x_{i'+1}) \geqslant d(x_{c'}, x_{i'})$. Thus, $f_{i,i'+1}(c'') = d(x_{c'}, x_{i'+1}$. With lemma 2, $d(x_{c'}, x_{i'+1} \geqslant d(x_c, x_{i'+1}$. $f_{i,i'}(c) = d(x_c, x_{i'})$ implies that $d(x_c, x_{i'}) \geqslant d(x_c, x_i)$, and then $d(x_c, x_i) \leqslant d(x_c, x_{i'}) \leqslant d(x_c, x_{i'+1})$. Thus $f_{i,i'+1}(c) = d(x_{c'}, x_{i'+1}$, and $f_{i,i'}(c) \leqslant f_{i,i'}(c')$. □

Proposition 6. *, Let $E = \{x_1, \dots, x_N\}$ be N points of $\mathbb{R}^2$, such that for all $i < j$, $x_i \prec x_j$. Algorithm 2 computes $f_{\mathcal{D}}(\mathcal{C}_{j,j'})^\alpha$ for all $j' \in [\![j; N]\!]$ for a given $j \in [\![1; N]\!]$ in $O(N)$ time using $O(N)$ memory space.*

Proof. The validity of Algorithm 2 is based on Lemmas 10 and 11: once a discrete center c is known for a $f_{\mathcal{D}}(\mathcal{C}_{j,j'})^\alpha$, we can find a center c' of $f_{\mathcal{D}}(\mathcal{C}_{j,j'+1})^\alpha$ with $c' \geqslant c$, and Lemma 10 gives the stopping criterion to prove a discrete center. Let us prove the time complexity; the space complexity is obviously within $O(N)$ memory space. In Algorithm 2, each computation $f_{j',j}(curCtr)$ is in $O(1)$ time; we have to count the number of calls for this function. In each loop in j', one computation is used for the initialization; the total number of calls for this initialization is $N - j \leqslant N$. Then, denoting, with $c_N \leqslant N$, the center found for $\mathcal{C}_{j,N}$, we note that the number of loops is $c_N - j \leqslant N$. Lastly, there are less that $2N$ computations calls $f_{j',j}(curCtr)$; Algorithm 2 runs in $O(N)$ time. □

Algorithm 2: Computing $f_{\mathcal{D}}(\mathcal{C}_{j,j'})^\alpha$ for all $j' \in [\![j; N]\!]$ for a given $j \in [\![1; N]\!]$

Input: $E = \{x_1, \dots, x_N\}$ indexed with Proposition 1, $j \in [\![1; N]\!]$, $\alpha > 0$, N points of $\mathbb{R}^2$,
Output: for all $j' \in [\![1; j]\!]$, $v_{j'} = f^\alpha_{\mathcal{D}}(\mathcal{C}_{j',j})$

```
define vector $v$ with $v_{j'} := 0$ for all $j' \in [\![j; N]\!]$
define $curCtr := j + 1$, $curCost := 0$
for $j' := j + 1$ to $N$
    $curCost := f_{j',j}(curCtr)$
    while $curCost \leqslant f_{j',j}(curCtr + 1)$
        $curCtr := curCtr + 1$
        $curCost := f_{j',j}(curCtr)$
    end while
    $v_{j'} := curCost^\alpha$
end for
return vector $v$
```

Proposition 7. *Let $E = \{x_1, \dots, x_N\}$ be N points of $\mathbb{R}^2$, such that for all $i < j$, $x_i \prec x_j$. Algorithm 3 computes $f_{\mathcal{D}}(\mathcal{C}_{j',j})^\alpha$ for all $j' \in [\![1; j]\!]$ for a given $j \in [\![1; N]\!]$ in $O(N)$ time, using $O(N)$ memory space.*

Algorithm 3: Computing $f_{\mathcal{D}}(\mathcal{C}_{j',j})^\alpha$ for all $j' \in [\![1;j]\!]$ for a given $j \in [\![1;N]\!]$

Input: $E = \{x_1, \dots, x_N\}$ indexed with Proposition 1, $j \in [\![1;N]\!]$, $\alpha > 0$, N points of $\mathbb{R}^2$,
Output: for all $j' \in [\![1;j]\!]$, $v_{j'} = f_{\mathcal{D}}(\mathcal{C}_{j',j})^\alpha$

```
    define vector $v$ with $v_{j'} := 0$ for all $j' \in [\![1;j]\!]$
    define $curCtr := j-1$, $curCost := 0$
    for $j' := j-1$ to 1 with increment $j' := j'-1$
        $curCost := f_{j',j}(curCtr)$
        while $curCost \leqslant f_{j',j}(curCtr-1)$
            $curCtr := curCtr - 1$
            $curCost := f_{j',j}(curCtr)$
        end while
        $v_{j'} := curCost^\alpha$
    end for
return vector $v$
```

Proof. The proof is analogous with Proposition 6, applied to Algorithm 2. □

5. Particular Sub-Cases

Some particular sub-cases have specific complexity results, which are presented in this section.

5.1. Sub-Cases with $K = 1$

We first note that sub-cases $K = 1$ show no difference between 1-0-+-(α, γ)-BC2DPF and 1-0-max-$(1, \gamma)$-BC2DPF problems, defining the continuous or the discrete version of 1-center problems. Similarly, 1-M-+-(α, γ)-BC2DPF and 1-M-max-$(1, \gamma)$-BC2DPF problems define the continuous or the discrete version of partial 1-center problems. 1-center optimization problems have a trivial solution; the unique partition of E in one subset is E. To solve the 1-center problem, it is necessary to compute the radius of the minimum enclosing disk covering all the points of E (centered in one point of E for the discrete version). Once the points are re-indexed with Proposition 1, the cost computation is in $O(1)$ time for the continuous version using Proposition 3, and in $O(\log N)$ time for the discrete version using Proposition 5. The cost of the re-indexation in $O(N \log N)$ forms the overall complexity time with such an approach. One may improve this complexity without re-indexing E.

Proposition 8. *Let $E = \{x_1, \dots, x_N\}$, a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \; \mathcal{I} \; x_j$. 1-0-$\oplus$-$(\alpha, \gamma)$-BC2DPF problems are solvable in $O(N)$ time using $O(1)$ additional memory space.*

Proof. Using Lemma 3 or Lemma 6, computations of f_γ are, at most, in $O(N)$ once the extreme elements following the order $\prec$ have been computed. Computation of the extreme points is also seen in $O(N)$, with one traversal of the elements of E, storing only the current minimal and maximal element with the order relation $\prec$. Finally, the complexity of one-center problems is in linear time. □

Proposition 9. *Let $M \in \mathbb{N}^*$, let $E = \{x_1, \dots, x_N\}$ a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \; \mathcal{I} \; x_j$. The continuous partial 1-center, i.e., 1-M-$\oplus$-$(\alpha, 0)$-BC2DPF problems, is solvable in $O(N \min(M, \log N))$ time. The discrete partial 1-center, i.e., 1-M-$\oplus$-$(\alpha, 1)$-BC2DPF problems, is solvable in $O(N \log N)$ time.*

Proof. Using Proposition 2, *one*-center problems are computed equivalently: $\min_{m \in [\![0;M]\!]} f_\gamma(\mathcal{C}_{1+m,N-m})^\alpha$.

For the continuous and the discrete case, re-indexing the whole PF with Proposition 1 runs in $O(N \log N)$ time, leading to M computations in $O(1)$ or $O(\log(N-M))$ time, which are dominated by the complexity of re-indexing. The time complexity for both cases are highest in $O(N \log N)$. In the continuous case, i.e., $\gamma = 0$, one requires only the M minimal and maximal points with the total order $\prec$ to compute the cluster costs using. If $M < \log N$, one may use one traversal of E, storing the current m minimal and extreme points, which has a complexity in $O(MN)$. Choosing among the two possible algorithms, the time complexity is in $O(N \min(M, \log N))$. $\square$

5.2. Sub-Cases with $K = 2$

Specific cases with $K = 2$ define two clusters, and one separation as defined in Proposition 2. For these cases, specific complexity results are provided, enumerating all the possible separations.

Proposition 10. *Let N points of $\mathbb{R}^2$, $E = \{x_1, \dots, x_N\}$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$.* 2-0-$\oplus$-*(α, γ)-BC2DPF problems are solvable in $O(N \log N)$ time, using $O(N^\gamma)$ additional memory space.*

Proof. Using Proposition 2, optimal solutions exist, considering two clusters: $\mathcal{C}_{1,i}$ and $\mathcal{C}_{i+1,N}$. One enumerates the possible separations $i \in [\![1; N]\!]$. First, the re-indexation phase runs in $O(N \log N)$ time, which will be the bottleneck for the time complexity. Enumerating the (N-1) values $f_\gamma(\mathcal{C}_{1,i})^\alpha \oplus f_\gamma(\mathcal{C}_{i+1,N})^\alpha$ and storing the minimal value induces (N-1) computations in $O(1)$ time for the continuous case $\gamma = 0$, and uses $O(1)$ additional memory space: the current best value and the corresponding index. Considering the discrete case, one uses $O(N)$ additional memory space $f_\gamma(\mathcal{C}_{1,i})^\alpha$, $f_\gamma(\mathcal{C}_{i+1,N})^\alpha$ to maintain the time complexity result. $\square$

One can extend the previous complexity results with the partial covering extension.

Proposition 11. *Let $E = \{x_1, \dots, x_N\}$ be a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$. 2-M-$\oplus$-(α, γ)-BC2DPF problems are solvable in $O(N((M+1)^2 + \log N))$ time and $O(N^\gamma)$ additional memory space, or in $O(N((M+1)^2 \log^\gamma N) + \log N))$ time and $O(1)$ additional memory space.*

Proof. After the re-indexation phase running in $O(\log N)$ time), Proposition 2 ensures that there is an optimal solution for 2-M-$\oplus$-(α, γ)-BC2DP, removing the $m_1 \geqslant 0$ first indexes, the $m_3 \geqslant 0$ last indexes, and $m_2 \geqslant 0$ points between the two selected clusters, with $m_1 + m_2 + m_3 \leqslant M$. Using Proposition 3, there is an optimal solution, exactly defining the M outliers, so that we can consider that $m_1 + m_2 + m_3 = M$. Denoting i as the last index of the first cluster, the first selected cluster is $\mathcal{C}_{1+m_1,i}$; the second one is $\mathcal{C}_{i+m_2+1,N-M+m_1+m_2}$. We have $i \geqslant m_1 + 1$ and $i + m_2 + 1 \leqslant N - M + m_1 + m_2$ i.e., $i \leqslant N - M + m_1$. We denote, with X, the following feasible i, m_1, m_2

$$X = \{(i, m_1, m_2) \in [\![1; N]\!] \times [\![0; M]\!]^2,\ \ 0 \leqslant m_1 + m_2 \leqslant M \ \text{ and } \ m_1 + 1 \leqslant i \leqslant N - M + m_1\}$$

Computing an optimal solution for 2-M-$\oplus$-(α, γ)-BC2DP brings the following enumeration

$$OPT = \min_{(i,m_1,m_2) \in X} f_\gamma(\mathcal{C}_{1+m_1,i})^\alpha + f_\gamma(\mathcal{C}_{i+m_2+1,N-M+m_1+m_2})^\alpha \tag{20}$$

In the continuous case (ie $\gamma = 0$), we use $O((M+1)^2)$ computations to enumerate the possible m_1, m_2, and $O(N)$ computations to enumerate the possible i once m_1, m_2 are defined. With cost computations running in $O(1)$ time, the computation of (20) by enumeration runs in $O(N(M+1)^2)$ time, after the re-indexation in $O(N \log N)$ time. This induces the time complexity announced for $\gamma = 0$. This computation uses $O(1)$ additional memory space, storing only the best current solution $(i, m_1, m_2) \in X$ and its cost; this is also the announced memory complexity .

In the discrete case (i.e., $\gamma = 1$), we use $O((M+1)^2)$ computations to enumerate the possible m_1, m_2, and $O(N)$ computations to enumerate the possible i once m_1, m_2 are fixed. This uses $O(1)$ additional memory space, and the total time complexity is $O(N \log N((M+1)^2)$. To decrease the time complexity, one can use two vectors of size N to store a given m_1, m_2, for which the cluster costs $f_\gamma(\mathcal{C}_{1+m_1,i})^\alpha$ and $f_\gamma(\mathcal{C}_{i+m_2+1,N-M+m_1+m_2})^\alpha$ are given by Algorithms 2 and 3, so that the total time complexity remains in $O(N((M+1)^2 + \log N))$ with an $O(N)$ additional memory space. These two variants, using $O(1)$ or $O(N)$ additional memory space, induce the time complexity announced in Proposition 11. □

5.3. Continuous Min-Sum K-Radii on A Line

To the best of our knowledge, the 1D continuous min-sum k-radii and the min-sum diameter problems were not previously studied. The specific properties hold, as proven in Lemma 12. This allows a time complexity of $O(N \log N)$.

Lemma 12. *Let $E = \{x_1, \dots, x_N\}$ be N points in a line of $\mathbb{R}^2$, indexed such that for all $i < j$, $x_i \prec x_j$. The min-sum k-radii in a line, K-0-+-(1,0)-BC2DPF, is equivalent to selecting the $K-1$ highest values of the distance among consecutive points, with the extremity of such segments defining the extreme points of the disks.*

Proof. Let a feasible and non-nested solution of K-0-+-$(1,0)$-BC2DPF be defined with clusters $\mathcal{C}_{a_1,b_1}, \mathcal{C}_{a_2,b_2}, \dots, \mathcal{C}_{a_K,b_K}$ such that $1 = a_1 \leqslant b_1 < a_2 \leqslant b_2 < \dots < a_K \leqslant b_K = N$. Using the alignment property, we can obtain

$$d(x_1, x_N) = \sum_{i=1}^{n-1} d(x_i, x_{i+1}) = \sum_{k=1}^{K} d(x_{a_k}, x_{b_k}) + \sum_{k=2}^{K} d(x_{b_{k-1}}, x_{a_k}) = \sum_{k=1}^{K} f_0(\mathcal{C}_{a_k,b_k}) + \sum_{k=2}^{K} d(x_{b_{k-1}}, x_{a_k})$$

Reciprocally, this is equivalent to considering K-0-+-$(1,0)$-BC2DPF or the maximization of the sum of $K-1$ sa a different distance among consecutive points. The latter problem is just computing the $K-1$ highest distances among consecutive points. □

Proposition 12. *Let $E = \{x_1, \dots, x_N\}$ be a subset of N points of $\mathbb{R}^2$ on a line. K-0-+-$(1,0)$-BC2DPF, the continuous min-sum-k-radii, is solvable in $O(N \log N)$ time and $O(N)$ memory space.*

Proof. Lemma 12 ensures the validity of Algorithm 4, determining the $K-1$ highest values of the distance among consecutive points. The additional memory space in Algorithm 4 is in O(N), computing the list of consecutive distances. Sorting the distances and the re-indexation both have a time complexity in $O(N \log N)$. □

Algorithm 4: Continuous min-sum K-radii on a line

```
Input: $K \in \mathbb{N}^*$, $N$ points of $\mathbb{R}^2$ on a line $E = \{x_1, \dots, x_N\}$

  re-index $E$ using Proposition 1
  initialize vector $v$ with $v_i := (i, d(x_{i+1}) - d(x_i))$ for $i \in [\![1; N-1]\!]$
  initialize vector $w$ with $v_j := 0$ for $j \in [\![1; K-1]\!]$
  sort vector $v$ with $d(x_{i+1}) - d(x_i)$ increasing
  for the $K-1$ elements of $v$ with the maximal value $d(x_{i+1}) - d(x_i)$, store the indexes $i$ in $w$
  sort $w$ in the increasing order
  initialize $\mathcal{P} = \emptyset$, $\underline{i} = \bar{i} = 1$, $OPT = 0$.
  for $j \in [\![1; K-1]\!]$ in the increasing order
    $\bar{i} := w_j$
    add $\mathcal{C}_{\underline{i},\bar{i}}$ in $\mathcal{P}$
    $OPT := OPT + f_{\mathcal{C}}(\mathcal{C}_{\underline{i},\bar{i}})$
    $\underline{i} := \bar{i} + 1$
  end for
  add $\mathcal{C}_{\underline{i},N}$ in $\mathcal{P}$
  $OPT := OPT + f_{\mathcal{C}}(\mathcal{C}_{\underline{i},N})$

return $OPT$ the optimal cost and the partition of selected clusters $\mathcal{P}$
```

6. Unified DP Algorithm and Complexity Results

Proposition 2 allows the design of a common DP algorithm for p-center problems and variants, and to prove polynomial complexities. The key element is to design Bellman equations.

Proposition 13 (Bellman equations)**.** *Defining $O_{i,k,m}$ as the optimal cost of k-m-$\oplus$-(α,γ)-BC2DPF among points $[\![1,i]\!]$ for all $i \in [\![1,N]\!]$, $k \in [\![1,K]\!]$ and $m \in [\![0,M]\!]$, we have the following induction relations*

$$\forall i \in [\![1,N]\!], \quad O_{i,1,0} = f_\gamma(\mathcal{C}_{1,i})^\alpha \tag{21}$$

$$\forall m \in [\![1,M]\!], \forall k \in [\![1,K]\!], \forall i \in [\![1,m+k]\!], \quad O_{i,k,m} = 0 \tag{22}$$

$$\forall m \in [\![1,M]\!], \forall i \in [\![m+2,N]\!], \quad O_{i,1,m} = \min(O_{i-1,1,m-1}, f_\gamma(\mathcal{C}_{1+m,i})^\alpha) \tag{23}$$

$$\forall k \in [\![2,K]\!], \forall i \in [\![k+1,N]\!], \quad O_{i,k,0} = \min_{j \in [\![k,i]\!]} \left(O_{j-1,k-1,0} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right) \tag{24}$$

$\forall m \in [\![1,M]\!]$, $\forall k \in [\![2,K]\!]$, $\forall i \in [\![k+m+1,N]\!]$,

$$O_{i,k,m} = \min\left(O_{i-1,k,m-1}, \min_{j \in [\![k+m,i]\!]} \left(O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right) \right) \tag{25}$$

Proof. (21) is the standard 1-center case. (22) is a trivial case, where it is possible to fill the clusters with singletons, with a null and optimal cost. (23) is a recursion formula among the partial 1-center cases, an optimal solution of 1-m-$\oplus$-(α,γ)-BC2DPF among points $[\![1,i]\!]$, selecting the point x_i, and the optimal solution is cluster $\mathcal{C}_{1+m,i}$ with Proposition 3, with a cost $f_\gamma(\mathcal{C}_{1+m,i})^\alpha$ or an optimal solution of 1-$m-1$-$\oplus$-(α,γ)-BC2DPF if the point x_i is not selected. (24) is a recursion formula among the k-0-$\oplus$-(α,γ)-BC2DPF cases among points $[\![1,i]\!]$; when generalizing the ones from [10] for the powered sum-radii cases, the proof is similar. Let $k \in [\![2,K]\!]$ and $i \in [\![k+1,N]\!]$. Let $j \in [\![k,i]\!]$, when selecting an optimal solution of k-0-$\oplus$-(α,γ)-BC2DPF among points indexed in $[\![1,j-1]\!]$, and adding cluster $\mathcal{C}_{j,i}$, a feasible solution is obtained for k-0-$\oplus$-(α,γ)-BC2DPF among the points indexed in

$[\![1,i]\!]$ with a cost $O_{j-1,k-1,0} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha$. This last cost is lower than the optimal cost, thus $O_{i,k,0} \leqslant O_{j-1,k-1,0} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha$. Such inequalities are valid for all $j \in [\![k,i]\!]$; this implies

$$O_{i,k,0} \leqslant \min_{j\in[\![k,i]\!]} \left(O_{j-1,k-1,0} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right) \tag{26}$$

Let $j_1 < j_2 < \cdots < j_{k-1} < j_k = i$ indexes, such that $\mathcal{C}_{1,j_1}, \mathcal{C}_{j_1+1,j_2}, \ldots, \mathcal{C}_{j_{k-1}+1,i}$ defines the optimal solution of k-0-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,i]\!]$; its cost is $O_{i,k,0}$. Necessarily, $\mathcal{C}_{1,j_1}, \mathcal{C}_{j_1+1,j_2}, \ldots, \mathcal{C}_{j_{k-2}+1,j_{k-1}}$ defines the optimal solution of $k-1$-0-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,j_{k-1}]\!]$. On the contrary, a better solution for $O_{i,k,0}$ would be constructed, adding the cluster $\mathcal{C}_{j_{k-1}+1,i}$. We thus have $O_{i,k,0} = O_{j_{k-1},k-1,0} \oplus f_\gamma(\mathcal{C}_{j_{k-1}+1,i})^\alpha$. Combined with (26), this proves $O_{i,k,0} = \min_{j\in[\![k,i]\!]} \left(O_{j-1,k-1,0} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right)$.

Lastly, we prove (25). Let $m \in [\![1,M]\!], k \in [\![2,K]\!], i \in [\![k+m+1,N]\!]$. $O_{i,k,m} \leqslant O_{i-1,k,m-1}$; each solution of k-$m-1$-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,i-1]\!]$ defines a solution of k-m-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,i]\!]$, with the selecting point x_i as an outlier. Let $O'_{i,k,m}$, with the cost of k-m-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,i]\!]$; necessarily selecting the point i, we obtain $O_{i,k,m} \leqslant O'_{i,k,m}$. $O'_{i,k,m}$ is defined by a cluster $\mathcal{C}_{j,i}$ and an optimal solution of k-m-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,j-1]\!]$, so that $O'_{i,k,m} = \min_{j\in[\![k+m,i]\!]} \left(O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right)$. We thus have

$$O_{i,k,m} \leqslant \min\left(O_{i-1,k,m-1}, \min_{j\in[\![k+m,i]\!]} \left(O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right) \right) \tag{27}$$

Reciprocally, let $1 = a_1 < b_1 < a_2 < b_2 < \cdots < a_k < b_k$ indexes, such that $\mathcal{C}_{a_1,b_1}, \mathcal{C}_{a_2,b_2}, \ldots, \mathcal{C}_{a_k,b_k}$ defines an optimal solution of k-m-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,i]\!]$; its cost is $O_{i,k,m}$. If $b_k = i$, then $O_{i,k,m} = O'_{i,k,m}$ and (27) is an equality. If $b_k < i$, then $\mathcal{C}_{a_1,b_1}, \mathcal{C}_{a_2,b_2}, \ldots, \mathcal{C}_{a_k,b_k}$ defines an optimal solution of k-$m-1$-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1,i-1]\!]$; its cost is $O_{i,k,m-1}$. We thus have $O_{i,k,m} = O_{i,k,m-1}$, and (27) is an equality. Finally, (25) is proven by disjunction. □

Bellman equations of Proposition 13 can compute the optimal value $O_{i,k,m}$ by induction. A first method is a recursive implementation of the Bellman equations to compute the cost $O_{N,K,M}$ and store the intermediate computations $O_{i,k,m}$ in a memoized implementation. An iterative implementation is provided in Algorithm 5, using a defined order for the computations of elements $O_{i,k,m}$. An advantage of Algorithm 5 is that independent computations are highlighted for a parallel implementation. For both methods computing the optimal cost $O_{N,K,M}$, backtracking operations in the DP matrix with computed costs allow for recovery of the affectation of clusters and outliers in an optimal solution.

In Algorithm 5, note that some useless computations are not processed. When having to compute $O_{N,K,M}$, computations $O_{N,k,m}$ with $k+m < K+M$ are useless. $O_{N-1,K,M}$ will also not be called. Generally, triangular elements $O_{N-n,k,m}$ with $n+k+m < K+M$ are useless. The DP matrix $O_{n,k,m}$ is not fully constructed in Algorithm 5, removing such useless elements.

Algorithm 5: unified DP algorithm for K-M-$\oplus$-(α,γ)-BC2DPF

```
Input:  - $N$ points of $\mathbb{R}^2$, $E = \{x_1, \dots, x_N\}$ such that for all $i \neq j$, $x_i \ \mathcal{I} \ x_j$ ;
        - Parameters: $K \in \mathbb{N}^*$, $M \in \mathbb{N}$, $\oplus \in \{+, \max\}$, $\gamma \in \{0,1\}$ and $\alpha > 0$ ;

  sort $E$ following the order of Proposition 1
  initialize matrix $O$ with $O_{i,k,m} := 0$ for all $m \in [\![0; M]\!], k \in [\![1; K-1]\!]$,
$i \in [\![k; N-K+k]\!]$

  compute $f_\gamma(\mathcal{C}_{1,i})^\alpha$ for all $i \in [\![1; N-K+1]\!]$ and store in $O_{i,1,0} := f_\gamma(\mathcal{C}_{1,i})^\alpha$

  for $i := 2$ to $N$
    compute and store $f_\gamma(\mathcal{C}_{i',i})^\alpha$ for all $i' \in [\![1; i]\!]$
    compute $O_{i,k,0} := \min_{j \in [\![k,i]\!]} \left( O_{j-1,k-1,0} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right)$ for all $k \in [\![2; \min(K,i)]\!]$
    for $m = 1$ to $\min(M, i-2)$
      compute $O_{i,1,m} := \min(O_{i-1,1,m-1}, f_\gamma(\mathcal{C}_{1+m,i})^\alpha)$
      for $k = 2$ to $\min(K, i-m)$
        compute $O_{i,k,m} := \min\left( O_{i-1,k,m-1}, \min_{j \in [\![k+m,i]\!]} \left( O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha \right) \right)$
      end for
    end for
    delete the stored $f_\gamma(\mathcal{C}_{i',i})^\alpha$ for all $i' \in [\![1; i]\!]$
  end for

  initialize $\mathcal{P} = \emptyset$, $\underline{i} = \bar{i} = N$, $m = M$
  for $k = K$ to 1 with increment $k \leftarrow k-1$
    compute $\bar{i} := \min\{i \in [\![\underline{i} - m; \underline{i}]\!] | O_{\underline{i},k,m} := O_{\underline{i}-i,k,m-i+\underline{i}}\}$
    $m := m - \bar{i} + \underline{i}$
    compute and store $f_\alpha(\mathcal{C}_{i',\bar{i}})$ for all $i' \in [\![1; \bar{i}]\!]$
    find $\underline{i} \in [\![1, \bar{i}]\!]$ such that $\underline{i} := \arg\min_{j \in [\![k+m,i]\!]} \left( O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,\bar{i}})^\alpha \right)$
    add $\mathcal{C}_{\underline{i},\bar{i}}$ in $\mathcal{P}$
    delete the stored $f_\alpha(\mathcal{C}_{i',\bar{i}})$ for all $i' \in [\![1; \bar{i}]\!]$
  end for

return $O_{N,K,M}$ the optimal cost and the selected clusters $\mathcal{P}$
```

Theorem 1. *Let $E = \{x_1, \dots, x_N\}$ a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$. When applied to the 2D PF E for $K \geqslant 2$, the K-M-$\oplus$-(α,γ)-BC2DPF problems are solvable to optimality in polynomial time using Algorithm 5, with a complexity in $O(KN^2(1+M))$ time and $O(KN(1+M))$ space.*

Proof. The validity of Algorithm 5 is proven by induction; each cell of the DP matrix $O_{i,k,m}$ is computed using only cells that were previously computed to optimality. Once the required cells are computed, a standard backtracking algorithm is applied to compute the clusters. Let us analyze the complexity. Let $K \geqslant 2$. The space complexity is in $O(KN(1+M))$, along with the size of the DP matrix, with the intermediate computations of cluster costs using, at most, O(N) memory space, only remembering such vectors due to the deleting operations. Let us analyze the time complexity. Sorting and indexing the elements of E (Proposition 1) has a time complexity in $O(N \log N)$. Once costs $f_\gamma(\mathcal{C}_{i',i})^\alpha$ are computed and stored, each cell of the DP matrix is computed, at most, in $O(N)$ time using Formulas (21)–(24). This induces a total complexity in $O(KN^2(1+M))$ time. The cluster costs are computed using N times Algorithm 3 and one time Algorithm 2; this has a time complexity in $O(N^2)$, which is negligible compared to the $O(KN^2(1+M))$ time computation of the cells of the DP matrix. The K backtracking operations requires a $O(N^2)$ time computation of the costs $f_\alpha(\mathcal{C}_{i',\bar{i}})$ for all $i' \in [\![1; \bar{i}]\!]$ and a given

i, M operations in $O(1)$ time to compute $\min\{i \in [\![\underline{i} - m; \underline{i}]\!] | O_{\underline{i},k,m} = O_{\underline{i}-i,k,m-i+\underline{i}}\}$ and $O(N)$ operations in $O(1)$ time to compute $\arg\min_{j \in [\![k+m,i]\!]} \left(O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,\bar{i}})^\alpha\right)$. Finally, the backtracking operations requires $O(KN^2)$ time, which is negligible compared to the previous computation in $O(KN^2(1+M))$ time. □

7. Specific Improvements

This section investigates how the complexity results of Theorem 2 may be improved, and how to speed up Algorithm 5, from a theoretical and a practical viewpoint.

7.1. Improving Time Complexity for Standard and Partial P-Center Problems

In Algorithm 5, the bottleneck for complexity are the computations $\min_{j \in [\![k+m,i]\!]} \left(O_{j-1,k-1,m} \oplus f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$, for $i \in [\![2, N]\!]$, $k \in [\![2, \min(K, i)]\!]$, $m \in [\![0, i-k]\!]$. When $\oplus = \max$, it is proven that such a minimization can be processed in $O(\log N)$ instead of $O(N)$ for the naive enumeration, leading to the general complexity results. This can improve the time complexity in the p-center cases.

Lemma 13. *Let $k \in [\![1, K]\!]$ and $j \in [\![1, N]\!]$. The application $m \in [\![0, M]\!] \longmapsto O_{j,k,m}$ is decreasing.*

Proof. Let $m \in\in [\![1, M]\!]$. For each $E' \subset E$, any feasible solution of k-$(m-1)$-$\oplus$-(α,γ)-BC2DPF in E' is a feasible solution of k-m-$\oplus$-(α,γ)-BC2DPF, with the partial versions defined by problems (11). An optimal solution of k-$m-1$-$\oplus$-(α,γ)-BC2DPF is feasible for k-$(m-1)$-$\oplus$-(α,γ)-BC2DPF, it implies $O_{j,k,m-1} \geqslant O_{j,k,m}$. □

Lemma 14. *Let $k \in [\![1, K]\!]$ and $m \in [\![0, M]\!]$. The application $j \in [\![1, N]\!] \longmapsto O_{j,k,m}$ is increasing.*

Proof. We yfirst note that the case $k = 1$ is implied by the Lemma 7, so that we can suppose in the following, that $k \geqslant 2$. Let $k \in [\![2, K]\!]$, $m \in [\![0, M]\!]$ and $j \in [\![2, N]\!]$. Let $\pi \in \Pi_K(E)$ be an optimal solution of k-m-$\oplus$-(α,γ)-BC2DPF among the points indexed in $[\![1, j]\!]$; its cost is $O_{j,k,m}$. Let $X \subset E$, the subset of the non-selected points, $|X| \leqslant M$, and $\mathcal{C}_1, \dots, \mathcal{C}_k$ with the k subsets defining the costs, so that $X, \mathcal{C}_1, \dots, \mathcal{C}_k$ is a partition of E and $\bigoplus_{k'=1}^{k} f_\gamma(\mathcal{C}_{k'})^\alpha = O_{j,k,m}$. If $x_j \in X$, then $O_{j,k,m} = O_{j-1,k,m-1} \geqslant O_{j-1,k,m}$ using Lemma 13, which is the result. We suppose to end the proof that $x_j \notin X$ and re-index the clusters such that $x_j \in \mathcal{C}_k$. We consider the clusters $\mathcal{C}'_1, \dots, \mathcal{C}'_k = \mathcal{C}_1, \dots, \mathcal{C}_{k-1}, \mathcal{C}_k - x_k$. With X, a partition of $(x_l)_{l \in [\![1, j-1]\!]}$ is defined, with, at most, M outliers, so that it defines a feasible solution of the optimization problem, defining $O_{j-1,k,m}$ as a cost $OPT' \geqslant O_{j-1,k,m}$. Using Lemma 7, $OPT' \leqslant O_{j,k,m}$, so that $O_{j-1,k,m-1} \geqslant O_{j-1,k,m}$. □

Lemma 15. *Let $i \in [\![2, N]\!]$, $k \in [\![2, \min(K, i)]\!]$, $m \in [\![0, i-k]\!]$. Let $g_{i,k,m} : j \in [\![2, i]\!] \longmapsto \max(O_{j-1,k-1,m}, f_\gamma(\mathcal{C}_{j,i})^\alpha)$. There is $l \in [\![2, i]\!]$, such that $g_{i,k}$ is decreasing for $j \in [\![2, l]\!]$, and then increases for $j \in [\![l+1, i]\!]$. For $j < l$, $g_{i,k} = f_\gamma(\mathcal{C}_{j,i})^\alpha$ and for $j > l$, $g_{i,k} = O_{j-1,k-1,m}$.*

Proof. Similarly to the proof of Lemma 10, the following applications are monotone:

$j \in [\![1, i]\!] \longmapsto f_\gamma(\mathcal{C}_{j,i})^\alpha$ decreases with Lemma 7,

$j \in [\![1, N]\!] \longmapsto O_{j,k,m}$ increases for all k with Lemma 14. □

Proposition 14. *Let $i \in [\![2, N]\!], k \in [\![2, K]\!], m \in [\![0, M]\!]$. Let $\gamma \in \{0, 1\}$. Once the values $O_{i',k-1,m'}$ in the DP matrix of Algorithm 2 are computed, Algorithm 6 computes $O_{i,k,m} = \min_{j \in [\![k+m,i]\!]} \max\left(O_{j-1,k-1,m}, f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$ calling $O(\log i)$ cost computations $f_\gamma(\mathcal{C}_{j,i})$. This induces a time complexity in $O(\log^{1+\gamma} i)$ using straightforward computations of the cluster costs with Propositions 3 and 5.*

Algorithm 6: Dichotomic computation of $\min_{j \in [\![k+m,i]\!]} \max\left(O_{j-1,k-1,m}, f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$

```
input: indexes $i \in [\![2,N]\!]$, $k \in [\![2,\min(K,i)]\!]$, $m \in [\![0,i-k]\!]$, $\alpha > 0$ $\gamma \in \{0,1\}$;
    a vector $v$ containing $v_j := O_{j,k-1,m}$ for all $j \in [\![1,i-1]\!]$.

define $\underline{i} := k+m$, $\underline{v} = f_\gamma(\mathcal{C}_{k+m,i})^\alpha$,
define $\overline{i} := i$, $\overline{v} := v_{i-1}$,
while $\overline{i} - \underline{i} \geqslant 2$
    $i'' := \left\lfloor \frac{\underline{i}+\overline{i}}{2} \right\rfloor$
    if $f_\gamma(\mathcal{C}_{j,i})^\alpha < v_{i''}$ then set $\overline{i} := i''$ and $\overline{v} := v_{i''}$
    else $\underline{i} := i''$ and $\underline{v} := f_\gamma(\mathcal{C}_{j,i})^\alpha$
end while
return $\min(\underline{v}, \overline{v})$
```

Proof. Algorithm 6 is a dichotomic search based on Lemma 15, similarly to Algorithm 1, derived from Lemma 10. The complexity in Algorithm 6 is $O(\log i)$ cost computations $f_\gamma(\mathcal{C}_{j,i})$. In the discrete case, such computations run in $O(\log i)$ time with Proposition 5, whereas it is $O(1)$ in the continuous case with Lemma 3. In both cases, the final time complexity is given by $O(\log^{1+\gamma} i)$. □

Computing $\min_{j \in [\![k+m,i]\!]} \max\left(O_{j-1,k-1,m}, f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$ in time $O(\log i)$ instead of $O(i)$ in the proof of Theorem 1 for p-center problem and variants, the complexity results are updated for these sub-problems.

Theorem 2. *Let $E = \{x_1, \dots, x_N\}$ be a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$. Whe napplied to the 2D PF E for $K \geqslant 2$, the K-M-max-(α,γ)-BC2DPF problems are solvable to optimality in polynomial time using Algorithm 4, with a complexity in $O(KN(1+M)\log N)$ time and $O(KN(1+M))$ space.*

Proof. The validity of Algorithm 5 using Algorithm 6 inside is implied by the validity of Algorithm 6, proven in Proposition 14. Updating the time complexity with Proposition 14, the new time complexity for continuous K-center problems is seen in $O(KN(1+M)\log N)$ time instead of $O(K(1+M)N^2)$, as previously. For the discrete versions, using Proposition 14 with computations of discrete cluster costs with Proposition 5 induces a time complexity in $O(KN(1+M)\log^2 N)$. The complexity is decreased to $O(KN(1+M)\log N)$, where the cluster costs are already computed and stored in Algorithm 5, and thus the computations of Algorithm 6 are seen in $O(1)$. tThisinduces the same complexity for discrete and continuous K-center variants. □

Remark 2. *For the standard discrete p-center, Theorem 2 improves the time complexity given in the preliminary paper [10], from $O(pN\log^2 N)$ to $O(pN\log N)$. Another improvement was given by Algorithm 1; the former computation of cluster costs has the same asymptotic complexity but requires two times more computations. tTis proportional factor is non negligible in practice.*

7.2. Improving Space Complexity for Standard P-Center Problems

For standard p-center problems, Algorithm 5 has a complexity in memory space in $O(KN)$, the size of the DP matrix. This section proves it is possible to reduce the space complexity into an $O(N)$ memory space.

One can compute the DP matrix for k-centers "line-by-line", with k increasing. This does not change the validity of the algorithm, with each computation using values that were previously computed to the optimal values. Two main differences occur compared to Algorithm 5. On one hand, the $k+1$-center values use only k-center computations, and the computations with $k' < k$ can be deleted once all the required k-center values are computed when having to compute only the K-center values, especially the optimal cost. On the other hand, the computations of cluster costs are not factorized, as in Algorithm 5; this does not make any difference in the continuous version, where Lemma 3 can to recompute cluster costs in $O(1)$ time when needed, whereas recomputing each cost induces the computations running in $O(\log N)$ for the discrete version with Algorithm 1.

The search order of operations slightly degrades the time complexity for the discrete variant, without inducing a change in the continuous variant. This allows only for computations of the optimal value; another difficulty is that the backtracking operations, as written in Algorithm 5, require storage of the whole stored values of the whole matrix. The issue is obtaining alternative backtracking algorithms that allow the computation of an optimal solution of the standard p-center problems using only the optimal value provided by the DP iterations, and with a complexity of, at most, $O(KN\log^{\gamma} N)$ time and $O(N)$ memory space. Algorithms 7 and 8 have such properties.

Algorithm 7: Backtracking algorithm using $O(N)$ memory space

```
    input: - $\gamma \in \{0,1\}$ to specify the clustering measure;
      - $N$ points of a 2D PF, $E = \{z_1, \dots, z_N\}$, sorted such that for all $i < j$, $z_i \prec z_j$ ;
      - $K \in \mathbb{N}$ the number of clusters;
      - $OPT$, the optimal cost of $K$-$\gamma$-CP2DPF;
    output: $\mathcal{P}$ an optimal partition of $K$-$\gamma$-CP2DPF.

    initialize $maxId := N$, $minId := N$, $\mathcal{P} = \emptyset$, a set of sub-intervals of $[\![1;N]\!]$.
    for $k := K$ to 2 with increment $k \leftarrow k-1$
      set $minId := maxId$
      while $f_\gamma(\mathcal{C}_{minId-1,maxId})) \leqslant OPT$ do $minId := minId - 1$ end while
      add $[\![minId, maxId]\!]$ in $\mathcal{P}$
      $maxId := minId - 1$
    end for
    add $[\![1, maxId]\!]$ in $\mathcal{P}$
return $\mathcal{P}$
```

Algorithm 8: Backtracking algorithm using $O(N)$ memory space

```
    input: - $\gamma \in \{0,1\}$ to specify the clustering measure;
      - $N$ points of a 2D PF, $E = \{z_1, \dots, z_N\}$, sorted such that for all $i < j$, $z_i \prec z_j$ ;
      - $K \in \mathbb{N}$ the number of clusters;
      - $OPT$, the optimal cost of $K$-$\gamma$-CP2DPF;
    output: $\mathcal{P}$ an optimal partition of $K$-$\gamma$-CP2DPF.

    initialize $minId := 1$, $maxId := 1$, $\mathcal{P} := \emptyset$, a set of sub-intervals of $[\![1;N]\!]$.
    for $k := 2$ to $K$ with increment $k \leftarrow k+1$
      set $maxId := minId$
      while $f_\gamma(\mathcal{C}_{minId,maxId+1})) \leqslant OPT$ do $maxId := maxId + 1$ end while
      add $[\![minId, maxId]\!]$ in $\mathcal{P}$
      set $minId := maxId + 1$
    end for
    add $[\![minId, N]\!]$ in $\mathcal{P}$
return $\mathcal{P}$
```

Lemma 16. *Let $K \in \mathbb{N}, K \geqslant 2$. Let $E = \{z_1, \dots, z_N\}$, sorted such that for all $i < j$, $z_i \prec z_j$. For the discrete and continuous K-center problems, the indexes given by Algorithm 7 are lower bounds of the indexes of any optimal solution. Denoting $[\![1, i_1]\!], [\![i_1+1, i_2]\!], \dots, [\![i_{K-1}+1, N]\!]$, the indexes given by Algorithm 7, and $[\![1, i'_1]\!], [\![i'_1+1, i'_2]\!], \dots, [\![i'_{K-1}+1, N]\!]$, the indexes of an optimal solution, we have, for all $k \in [\![1, K-1]\!], i_k \leqslant i'_k$*

Proof. This lemma is proven a decreasing induction on k, starting from $k = K-1$. The case $k = K-1$ is furnished by the first step of Algorithm 4, and $j \in [\![1,N]\!] \longmapsto f_\gamma(\mathcal{C}_{j,N})$ decreaswa with Lemma 7. WIth a given k, $i'_k \leqslant i_k$, $i_{k-1} \leqslant i'_{k-1}$ is implied by Lemma 2 and $d(z_{i_k}, z_{i_{k-1}-1}) > OPT$. □

Algorithm 8 is similar to Algorithm 7, with iterations increasing the indexes of the points of E. The validity is similarly proven, and this provides the upper bounds for the indexes of any optimal solution of K-center problems.

Lemma 17. *Let $K \in \mathbb{N}, K \geqslant 2$. Let $E = \{z_1, \dots, z_N\}$, sorted such that for all $i < j$, $z_i \prec z_j$. For K-center problems, the indexes given by Algorithm 8 are upper bounds of the indexes of any optimal solution. Denoting $[\![1,i_1]\!], [\![i_1+1,i_2]\!], \dots, [\![i_{K-1}+1,N]\!]$, the indexes given by Algorithm 8, and $[\![1,i'_1]\!], [\![i'_1+1,i'_2]\!], \dots, [\![i'_{K-1}+1,N]\!]$, the indexes of an optimal solution, we have, for all $k \in [\![1,K-1]\!]$, $i_k \geqslant i'_k$.*

Proposition 15. *Once the optimal cost of p-center problems are computed, Algorithms 7 and 8 compute an optimal partition in $O(N \log N)$ time using $O(1)$ additional memory space.*

Proof. We consider the proof for Algorithm 7, which is symmetrical for Algorithm 8. Let OPT be the optimal cost of K-center clustering with f. Let $[\![1,i_1]\!], [\![i_1+1,i_2]\!], \dots, [\![i_{K-1}+1,N]\!]$ be the indexes given by Algorithm 7. Through this construction, all the clusters $\mathcal{C}$ defined by the indexes $[\![i_k+1,i_{k+1}]\!]$ for all $k > 1$ verify $f_\gamma(\mathcal{C}) \leqslant OPT$. Let $\mathcal{C}_1$ be the cluster defined by $[\![1,i_1]\!]$; we have to prove that $f_\gamma(\mathcal{C}_1) \leqslant OPT$ to conclude the optimality of the clustering defined by Algorithm 4. For an optimal solution, let $[\![1,i'_1]\!], [\![i'_1+1,i'_2]\!], \dots, [\![i'_{K-1}+1,N]\!]$ be the indexes defining this solution. Lemma 16 ensures that $i_1 \leqslant i'_1$, and thus Lemma 7 assures $f_\gamma(\mathcal{C}_{1,i_1}) \leqslant f_\gamma(\mathcal{C}_{1,i'_1}) \leqslant OPT$. Analyzing the complexity, Algorithm 7 calls for a maximum of $(K+N) \leqslant 2N$ times the clustering cost function, without requiring stored elements; the complexity is in $O(N \log^\gamma N)$ time. □

Remark 3. *Finding the biggest cluster with an extremity given and a bounded cost can be acheived by a dichotomic search. Rhis would induce a complexity in $O(K \log^{1+\gamma} N)$. To avoid the separate case $K = O(N)$ and $\gamma = 1$, Algorithms 7 and 8 provide a common algorithm running in $O(N \log N)$ time, which is enough for the following complexity results.*

The previous improvements, written in Algorithm 9, allow for new complexity results with a $O(N)$ memory space for K-centrer problems.

Algorithm 9: p-center clustering in a 2DPF with a O(N) memory space

```
Input:
- $N$ points of $\mathbb{R}^2$, $E = \{x_1, \dots, x_N\}$ such that for all $i \neq j$, $x_i \; \mathcal{I} \; x_j$ ;
- $\gamma \in \{0,1\}$ to specify the clustering measure;
- $K \in \mathbb{N}$ the number of clusters.

  initialize matrix $O$ with $O_{i,k} := 0$ for all $i \in [\![1;N]\!], k \in [\![1;K-1]\!]$
  sort $E$ following the order of Proposition 1
  compute and store $O_{i,1} := f_\gamma(\mathcal{C}_{1,i})$ for all $i \in [\![1;N]\!]$ (with Algorithm 2 if $\gamma = 1$)
  for $k = 2$ to $K-1$
    for $i = k+1$ to $N-K+k$
      compute and store $O_{i,k} := \min_{j \in [\![2,i]\!]} \max(O_{j-1,k-1}, f_\gamma(\mathcal{C}_{j,i}))$ (Algorithm 6)
    end for
    delete the stored $O_{i,k-1}$ for all $i$
  end for
  $OPT := \min_{j \in [\![2,N]\!]} \max(O_{j-1,K-1}, f_\gamma(\mathcal{C}_{j,N}))$ with Algorithm 6
return $OPT$ the optimal cost and a partition $\mathcal{P}$ given by backtracking Algorithm 7 or 8
```

Theorem 3. *Let $E = \{x_1, \dots, x_N\}$ a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$. When applied to the 2D PF E for $K \geqslant 2$, the standard continuous and discrete K-center problems, i.e., K-*0-max-(α,γ)*-BC2DPF, are solvable with a complexity in $O(KN\log^{1+\gamma} N)$ time and $O(N)$ space.*

Remark 4. *The continuous case improves the complexity obtained after Theorem 2, with the same time complexity and an improvement in the space complexity. For the discrete variant, improving the space complexity in $O(N)$ instead of $O(N)$ induces a very slight degradation of the time complexity, from $O(KN\log N)$ to $O(KN\log^2 N)$. Depending on the value of K, it may be preferable, with stronger constraints in memory space, to have this second version.*

7.3. Improving Space Complexity for Partial P-Center Problems?

This section tries to generalize the previous results for the partial K-center problems, i.e., K-M-max-(α,γ)-BC2DPF with $M > 0$. The key element is to obtain a backtracking algorithm that does not use the DP matrix. Algorithm 10 extends Algorithm 7 by considering all the possible cardinals of outliers between clusters k and $k+1$ for $k \in [\![0, K-1]\!]$ and the outliers after the last cluster. A feasible solution of the optimal cost should be feasible by iterating Algorithm 7 for at least one of these sub-cases.

Algorithm 10: Backtracking algorithm for K-M-max-(α,γ)-BC2DPF with $M > 0$

```
    input: - a $K$-$M$-max-$(\alpha,\gamma)$-BC2DPF problem
     - $N$ points of a 2D PF, $E = \{z_1, \dots, z_N\}$, sorted such that for all $i < j$, $z_i \prec z_j$ ;
     - $OPT$, the optimal cost of $K$-$M$-max-$(\alpha,\gamma)$-BC2DPF problem;
    output: $\mathcal{P}$ an optimal partition of $K$-$M$-max-$(\alpha,\gamma)$-BC2DPF problem.

    for each vector $x$ of $K+1$ elements such that $\sum_{k=0}^{K} x[k] = M$
       initialize $maxId - x[K] := N$, $minId := N - x[K]$, $\mathcal{P} := \varnothing$, a set of sub-intervals
  of $[\![1; N]\!]$.
       for $k = K$ to 2 with increment $k \leftarrow k-1$
          set $minId := maxId$
          while $f_\gamma(\mathcal{C}_{minId-1,maxId}))^\alpha \leqslant OPT$ do $minId := minId - 1$ end while
          add $[\![minId, maxId]\!]$ in $\mathcal{P}$
          set $maxId := minId - 1 - x[K-1]$
       end for
       if $f_\gamma(\mathcal{C}_{1+x[0],maxId}))^\alpha \leqslant OPT$ then add $[\![1 + x[0], maxId]\!]$ in $\mathcal{P}$ and return $\mathcal{P}$
    end for
 return error "OPT is not a feasible cost for $K$-$M$-max-$(\alpha,\gamma)$-BC2DPF "
```

It is crucial to analyze the time complexity induced by this enumeration. If the number of vectors x of $K+1$ elements is such such that $\sum_{k=0}^{K} x[k] = M$ is in $\Theta(K^M)$, then this complexity is not polynomial anymore. For $M = 1$, a time complexity in $O(KN\log N)$ would be induced, which is acceptable within the complexity of the computation of the DP matrix. Having $M \geqslant 2$ would dramatically degrade the time complexity. Hence, we extend the improvement results of space complexity only for $M = 1$, with Algorithm 11.

Theorem 4. *Let $E = \{x_1, \dots, x_N\}$ a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$. When applied to the 2D PF E for $K \geqslant 2$, partial K-center problems* K-1-max-(α,γ)*-BC2DPF, are solvable with a complexity in $O(KN\log^{1+\gamma} N)$ time and $O(N)$ space.*

7.4. Speeding-Up DP for Sum-Radii Problems

Similarly to Algorithm 6, this section tries to speed up the computations $\min_{j \in [\![k+m,i]\!]}\left(O_{j-1,k-1,m} + f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$, which are the bottleneck for the time complexity in Algorithm 5. This section presents the stopping criterion to avoid useless computations in the $O(N)$ naive enumeration, but without providing proofs of time complexity improvements.

Algorithm 11: Partial p-center K-1-max-$(1,\gamma)$-BC2DPF with a O(N) memory space

```
Input:
- $N$ points of $\mathbb{R}^2$, $E = \{x_1, \dots, x_N\}$ such that for all $i \neq j$, $x_i \;\mathcal{I}\; x_j$ ;
- $\gamma \in \{0,1\}$ to specify the clustering measure;
- $K \in \mathbb{N}, K \geqslant 2$ the number of clusters.

    initialize matrix $O$ with $O_{i,k,m} := 0$ for all $i \in [\![1;N]\!], k \in [\![1;K-1]\!], m \in [\![0;1]\!]$
    sort $E$ following the order of Proposition 1
    compute and store $O_{i,1,0} := f_\gamma(\mathcal{C}_{1,i})$ for all $i \in [\![1;N]\!]$ (with Algorithm 2 if $\gamma = 1$)
    compute and store $f_\gamma(\mathcal{C}_{2,i})$ for all $i \in [\![2;N]\!]$ (with Algorithm 2 if $\gamma = 1$)
    compute and store $O_{i,1,1} := \min(f_\gamma(\mathcal{C}_{2,i}), O_{i-1,1,0})$ for all $i \in [\![2;N]\!]$
    for $k = 2$ to $K$
      for $i := k+1$ to $N-K+k$
        compute and store $O_{i,k,0} := \min_{j \in [\![2,i]\!]} \max(O_{j-1,k-1,0}, f_\gamma(\mathcal{C}_{j,i}))$ (Algorithm 6)
        compute and store $O_{i,k,1} := \min_{j \in [\![k+1,i]\!]} \max\left(O_{j-1,k-1,1}, f_\gamma(\mathcal{C}_{j,i})\right)$
        $O_{i,k,1} := \min(O_{i-1,k,0}, O_{i,k,1})$
      end for
      delete the stored $O_{i,k-1,m}$ for all $i,m$
    end for
return $O_{N,K,1}$ the optimal cost and a partition $\mathcal{P}$ given by backtracking Algorithm 10
```

Proposition 16. *Let $m \in [\![0,M]\!]$, $i \in [\![1,N]\!]$ and $k \in [\![2,K]\!]$. Let β an upper bound for $O_{i,k,m}$. We suppose there exist $j_0 \in [\![1,i]\!]$, such that $f_\gamma(\mathcal{C}_{j_0,i})^\alpha \geqslant \beta$. Then, each optimal index j^*, such that $O_{i,k,m} = O_{j^*-1,k-1,m} + f_\alpha(\mathcal{C}_{j^*,i})$ necessarily fulfills $j^* > j_0$. In other words, $O_{i,k,m} = \min_{j \in [\![\max(k+m,j_0+1),i]\!]}\left(O_{j-1,k-1,m} + f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$.*

Proof. With $f_\gamma(\mathcal{C}_{j_0,i})^\alpha \geqslant \beta$, Lemma 7 implies that for all $j < j_0$, $f_\gamma(\mathcal{C}_{j,i})^\alpha > f_\gamma(\mathcal{C}_{j_0,i})^\alpha \geqslant \beta$. Using $O_{i',k',m} \geqslant 0$ for all i', k' implies that for all $j < j_0$, $f_\gamma(\mathcal{C}_{j_0,i})^\alpha + O_{j_0-1,k-1,m} > \beta$, and the optimal index gives $O_{i,k,m} = \min_{j \in [\![k+m,i]\!]}\left(O_{j-1,k-1,m} + f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$, which is superior to j_0. □

Proposition 16 can be applied to compute each value of the DP matrix using fewer computations than the naive enumeration. In the enumeration, β is updated to the best current value of $\left(O_{j-1,k-1,m} + f_\gamma(\mathcal{C}_{j,i})^\alpha\right)$. The index would be enumerated in a decreasing way, starting from $j = i$ until an index is found, such that $f_\gamma(\mathcal{C}_{j_0,i})^\alpha \geqslant \beta$, and no more enumeration is required with Proposition 16, ensuring that the partial enumeration is sufficient to find the wished-for minimal value. This is a practical improvement, but we do not furnish proof of complexity improvements, as it is likely that this would not change the worst case complexity.

8. Discussion

8.1. Importance of the 2D PF Hypothesis, Summarizing Complexity Results

Planar p-center problems were not studied previously in the PF case. The 2D PF hypothesis is crucial for the complexity results and the efficiency of the solving algorithms. Table 1 compares the available complexity results for 1D and 2D cases of some k-center variants.

The complexity for 2D PF cases is very similar to the 1D cases; the 2D PF extension does not induce major difficulties in terms of complexity results. 2D PF cases may induce significant differences compared to the general 2D cases. The p-center problems are NP-hard in a planar Euclidean space [17], since adding the PF hypothesis leads to the polynomial complexity of Theorem 1, which allows for an efficient, straightforward implementation of the algorithm. Two properties of 2D PF were crucial for these results: The 1D structure implied by Proposition 1, which allows an extension of DP algorithms [58,59], and Lemmas 3 and 6, which allow quick computations of cluster costs. Note that rectangular p-center problems have a better complexity using general planar results than using our Theorems 2 and 3. Our algorithms only use common properties for Chebyshev and Minkowski distances, whereas significant improvements are provided using specificities of Chebyshev distance.

Table 1. Comparison of the time complexity for 2D PF cases to the 1D and 2D cases.

Problem	**1D Complexity**		**Our 2D PF Complexity**		**2D Complexity**	
Cont. min-sum-K-radii	$O(N \log N)$	Proposition 12	$O(KN^2)$	Theorem 1	NP-hard	[40]
Cont. p-center	$O(N \log^3 N)$	[7]	$O(pN \log N)$	Theorems 2 and 3	NP-hard	[17]
Discr. p-center	$O(N)$	[33]	$O(pN \log N)$	Theorem 2	NP-hard	[17]
Cont. 1-center	$O(N)$	[20]	$O(N)$	Proposition 8	$O(N)$	[20]
Discr. 1-center	-	-	$O(N)$	Proposition 8	$O(N \log N)$	[29]
Cont. 2-center	-	-	$O(N \log N)$	Proposition 10	$O(N \log^2 N)$	[28]
Discr. 2-center	-	-	$O(N \log N)$	Proposition 10	$O(N^{4/3} \log^5 N)$	[32]
partial 1-center	-	-	$O(N \min(M, \log N))$	Proposition 9	$O(N^2 \log N)$	[38]
Rect. 1-center	$O(N)$	[36]	$O(N)$	Proposition 2	$O(N)$	[36]
Rect. 2-center	$O(N)$	[36]	$O(N \log N)$	Proposition 10	$O(N)$	[36]
Cont. rect. p-center	$O(N)$	[36]	$O(pN \log N)$	Theorem 3	$O(N)$	[36]
Discr. rect. p-center	$O(N \log N)$	[36]	$O(pN \log N)$	Theorem 2	$O(N \log N)$	[36]

Note that our complexity results are given considering the complexity of the initial re-indexation with Proposition 1. This $O(N \log N)$ phase may be the bottleneck for the final complexity. Some papers mention results which consider that the data are already in the memory (avoiding an O(N) traversal for input data) and already sorted. In our applications, MOO methods such as epsilon-constraint provide already sorted points [3]. Using this means of calculating the complexity, our algorithms for continuous and discrete 2-center problems in a 2D PF would have, respectively, a complexity in $O(\log N)$ and $O(\log^2 N)$ time. A notable advantage of the specialized algorithm in a 2D PF instead of the general cases in 2D is the simple and easy to implement algorithms.

8.2. Equivalent Optimal Solutions for P-Center Problems

Lemmas 16 and 17 emphasize that many optimal solutions may exist; the lower and upper bounds may define a very large funnel. We also note that many optimal solutions can be nested, i.e., non-verifying the Proposition 2. For real-world applicationa, having well-balanced clusters is more natural, and often wished for. Algorithms 7 and 8 provide the most unbalanced solutions. One may balance the sizes of covering balls, or the number of points in the clusters. Both types of solutions may be given using simple and fast post-processing. For example, one may proceed with a steepest descent local search using two-center problem types for consecutive clusters in the current solution. For balancing the size of clusters, iterating two-center computations induces marginal computations in $O(\log^{1+\gamma} N)$ time for each iteration with Algorithm 6. Such complexity occurs once the points are re-indexed using Proposition 1; one such computation in $O(N \log N)$ allows for many neighborhood computations running in $O(\log^{1+\gamma} N)$ time, and the sorting time is amortized.

8.3. Towards a Parallel Implementation

Complexity issues are raised to speed-up the convergence of the algorithms in practice. An additional way to speed up the algorithms in practice is to consider implementation issues, especially parallel implementation properties in multi- or many-core environments. In Algorithm 5, the values of the DP matrix $O_{i,k,m}$ for a given $i \in [\![1; N]\!]$ requires only to compute the values $O_{j,k,m}$ for all $j < i$. Independent computations can thus be operated at the iteration i of Algorithm 5, once the cluster costs $f_\gamma(\mathcal{C}_{i',i})^\alpha$ for all $i' \in [\![1; i]\!]$ have been computed, which is not the most time-consuming part when using Algorithms 2 and 3. This is a very useful property for a parallel implementation, requiring only $N - 1$

synchronizations to process $O(KN^2(1+M))$ operations. Hence, a parallel implementation of Algorithm 5 is straightforward in a shared memory parallelization, using OpenMP for instance in C/C++, or higher-level programming languages such as Python, Julia or Chapel [60]. One may also consider an intensive parallelization in a many-core environment, such as General Purpose Graphical Processing Units (GPGPU). A difficulty when using this may be the large memory size that is required in Algorithm 5.

Section 7 variants, which construct the DP matrix faster, both for k-center and min-sum k-radii problems, are not compatible with an efficient GPGPU parallelization, and one would prefer the naive and fixed-size enumeration of Algorithm 5, even with its worse time complexity for the sequential algorithm. Comparing the sequential algorithm to the GPGPU parallelization, having many independent parallelized computations allows a huge proportional factor with GPGPU, which can compensate the worst asymptotic complexity for reasonable sized instances. Shared memory parallelization, such as OpenMP, is compatible with the improvements provided in Section 7. Contrary to Algorithm 5, Algorithms 9 and 11 compute the DP matrix with index k increasing, with $O(N)$ independent computation induced at each iteration. With such algorithms, there are only $K-2$ synchronizations required, instead of $N-1$ for Algorithm 5, which is a better property for parallelization. The $O(N)$ memory versions are also useful for GPGPU parallelization, where memory space is more constrained than when storing a DP matrix on the RAM.

Previously, the parallelization of the DP matrix construction was discussed, as this is the bottleneck in time complexity. The initial sorting algorithm can also be parallelized on GPGPU if needed; the sorting time is negligible in most cases. The backtracking algorithm is sequential to obtain clusters, but with a low complexity in general, so that a parallelization of this phase is not crucial. We note that there is only one case where the backtracking Algorithm has the same complexity as the construction of the DP matrix: the DP variant in $O(N)$ memory space proposed in Algorithm 11 with Algorithm 10 as a specific backtrack. In this specific case, the $O(K)$ tests with different positions of the chosen outlier are independent, which allows a specific parallelization for Algorithm 10.

8.4. Applications to Bi-Objective Meta-Heuristics

The initial motivation of this work was to support decision makers when an MOO approach without preference furnishes a large set of non-dominated solutions. In this application, the value of K is small, allowing for human analyses to offer some preferences. In this paper, the optimality is not required in the further developments. Our work can also be applied to a partial PF furnished by population meta-heuristics [5]. A posteriori, the complexity allows for the use of Algorithms 5, 9 and 11 inside MOO meta-heuristics. Archiving PF is a common issue of population meta-heuristics, facing multi-objective optimization problems [4,5]. A key issue is obtaining diversified points of the PF in the archive, to compute diversified solutions along the current PF.

Algorithms 5, 9 and 11 can be used to address this issue, embedded in MOO approaches, similarly to [49]. Archiving diversified solutions of Pareto sets has application for the diversification of genetic algorithms, to select diversified solutions for cross-over and mutation phases [61,62], but also for swarm particle optimization heuristics [63]. In these applications, clustering has to run quickly. The complexity results and the parallelization properties are useful in such applicationas.

For application to MOO meta-heuristics like evolutionary algorithms, the partial versions are particularly useful. Indeed, partial versions may detect outliers that are isolated from the other points. For such points, it is natural to process intensification operators to look for efficient solutions in the neighborhood, which will make the former outlier less isolated. Such a process is interesting for obtaining a better balanced distribution of the points along the PF, which is a crucial point when dealing with MOO meta-heuristics.

8.5. How to Choose K, M?

A crucial point in clustering applications the selection of an appropriate value of K. A too-small value of K can miss that instances which are well-captured with $K + 1$ representative clusters. Real-world applications seek the best compromise between the minimization of K, and the minimization of the dissimilarity among the clusters. Similarly, with [11], the properties of DP can be used to achieve this goal. With the DP Algorithm 9, many couples $\{(k, O_{N,k})\}_k$ are computed, using the optimal k-center values with k clusters. Having defined a maximal value K', the complexity for computing these points is seen in $O(NK' \log^{1+\gamma} N)$. When searching for good values of k, the elbow technique, may be applied. Backtracking operations may be used for many solutions without changing the complexity. Rhe same ideas are applicable along the M index. In the previoulsy described context of MOO meta-heuristics, the sensitivity with the M parameter is more important than the sensitivity for the parameter K, where the number of archived points is known and fixed regarding other considerations, such as the allowed size of the population.

9. Conclusions and Perspectives

This paper examined the properties of p-center problems and variants in the special case of a discrete set of non-dominated points in a 2D space, using Euclidean, Minkowski or Chebyshev distances. A common characterization of optimal clusters is proven for the discrete and continuous variants of the p-center problems and variants. Thie can solve these problems to optimality with a unified DP algorithm of a polynomial complexity. Some complexity results for the 2D PF case improve the general ones in 2D. The presented algorithms are useful for MOO approaches. The complexity results, in $O(KN \log N)$ time for the standard K-center problems, and in $O(KN^2)$ time for the standard min-sum k-radii problems, are useful for application with a large PF. When applied to N points and able to ncover $M < N$ points, partial K-center and min-sum-K-radii variants are, respectively, solvable in $O(K(M+1)N \log N)$ and $O(K(M+1)N^2)$ time. Furthermore, the DP algorithms have interesting properties for efficient parallel implementation in a shared memory environment, such as OpenMP or using GPGPU. This allows their application for a very large PF with short solving times. For an application for MOO meta-heuristics such as evolutionary algorithms, the partial versions are useful for the detection of outliers where intensification phases around these isolated solutions may be processed in order to obtain a better distribution of the points along the PF.

Future perspectives include the extension of these results to other clustering algorithms. The weighted versions of p-center variants were not studied in this paper, which was motivated by MOO perspectives, and future perspectives shall consider extending our algorithms to weighted variants. Regarding MOO applications, extending the results to dimension 3 is a subject of interest for MOO problems with three objectives. However, clustering a 3D PF will be an NP-hard problem as soon as the general 2D cases are proven to be NP-hard. The perspective in such cases is to design specific approximation algorithms for a 3D PF.

Author Contributions: Conceptualization, N.D. and F.N.; Methodology, N.D. and F.N.; Validation, E.-G.T. and F.N.; Writing–original draft preparation, N.D.; Writing—review and editing, N.D.; Supervision, E.-G.T. and F.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Peugeot, T.; Dupin, N.; Sembely, M.J.; Dubecq, C. MBSE, PLM, MIP and Robust Optimization for System of Systems Management, Application to SCCOA French Air Defense Program. In *Complex Systems Design & Management*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 29–40. [CrossRef]
2. Dupin, N.; Talbi, E. Matheuristics to optimize refueling and maintenance planning of nuclear power plants. *J. Heuristics* **2020**, 1–43. [CrossRef]
3. Ehrgott, M.; Gandibleux, X. Multiobjective combinatorial optimization-theory, methodology, and applications. In *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 369–444.
4. Schuetze, O.; Hernandez, C.; Talbi, E.; Sun, J.; Naranjani, Y.; Xiong, F. Archivers for the representation of the set of approximate solutions for MOPs. *J. Heuristics* **2019**, *25*, 71–105. [CrossRef]
5. Talbi, E. *Metaheuristics: From Design to Implementation*; Wiley: Hoboken, NJ, USA, 2009; Volume 74.
6. Hsu, W.; Nemhauser, G. Easy and hard bottleneck location problems. *Discret. Appl. Math.* **1979**, *1*, 209–215. [CrossRef]
7. Megiddo, N.; Tamir, A. New results on the complexity of p-centre problems. *SIAM J. Comput.* **1983**, *12*, 751–758. [CrossRef]
8. Ravi, S.; Rosenkrantz, D.; Tayi, G. Heuristic and special case algorithms for dispersion problems. *Oper. Res.* **1994**, *42*, 299–310. [CrossRef]
9. Wang, D.; Kuo, Y. A study on two geometric location problems. *Inf. Process. Lett.* **1988**, *28*, 281–286. [CrossRef]
10. Dupin, N.; Nielsen, F.; Talbi, E. Clustering a 2d Pareto Front: P-center problems are solvable in polynomial time. In Proceedings of the International Conference on Optimization and Learning, Cádiz, Spain, 17–19 February 2020; pp. 179–191. [CrossRef]
11. Dupin, N.; Nielsen, F.; Talbi, E. k-medoids clustering is solvable in polynomial time for a 2d Pareto front. In Proceedings of the World Congress on Global Optimization, Metz, France, 8–10 July 2019; pp. 790–799. [CrossRef]
12. Borzsony, S.; Kossmann, D.; Stocker, K. The skyline operator. In Proceedings of the 17th International Conference on Data Engineering, Heidelberg, Germany, 2–6 April 2001; pp. 421–430.
13. Nielsen, F. Output-sensitive peeling of convex and maximal layers. *Inf. Process. Lett.* **1996**, *59*, 255–259. [CrossRef]
14. Arana-Jiménez, M.; Sánchez-Gil, C. On generating the set of nondominated solutions of a linear programming problem with parameterized fuzzy numbers. *J. Glob. Optim.* **2020**, *77*, 27–52. [CrossRef]
15. Daskin, M.; Owen, S. Two New Location Covering Problems: The Partial P-Center Problem and the Partial Set Covering Problem. *Geogr. Anal.* **1999**, *31*, 217–235. [CrossRef]
16. Calik, H.; Labbé, M.; Yaman, H. p-Center problems. In *Location Science*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 79–92.
17. Megiddo, N.; Supowit, K. On the complexity of some common geometric location problems. *SIAM J. Comput.* **1984**, *13*, 182–196. [CrossRef]
18. Hochbaum, D. When are NP-hard location problems easy? *Ann. Oper. Res.* **1984**, *1*, 201–214. [CrossRef]
19. Hochbaum, D.; Shmoys, D. A best possible heuristic for the k-center problem. *Math. Oper. Res.* **1985**, *10*, 180–184. [CrossRef]
20. Gonzalez, T. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.* **1985**, *38*, 293–306. [CrossRef]
21. Daskin, M. *Network and Discrete Location: Models, Algorithms and Applications*; Wiley: Hoboken, NJ, USA, 1995.
22. Calik, H.; Tansel, B. Double bound method for solving the p-center location problem. *Comput. Oper. Res.* **2013**, *40*, 2991–2999. [CrossRef]
23. Elloumi, S.; Labbé, M.; Pochet, Y. A new formulation and resolution method for the p-center problem. *INFORMS J. Comput.* **2004**, *16*, 84–94. [CrossRef]
24. Callaghan, B.; Salhi, S.; Nagy, G. Speeding up the optimal method of Drezner for the p-centre problem in the plane. *Eur. J. Oper. Res.* **2017**, *257*, 722–734. [CrossRef]
25. Drezner, Z. The p-centre problem—heuristic and optimal algorithms. *J. Oper. Res. Soc.* **1984**, *35*, 741–748.
26. Hwang, R.; Lee, R.; Chang, R. The slab dividing approach to solve the Euclidean P-Center problem. *Algorithmica* **1993**, *9*, 1–22. [CrossRef]
27. Agarwal, P.; Procopiuc, C. Exact and approximation algorithms for clustering. *Algorithmica* **2002**, *33*, 201–226. [CrossRef]
28. Megiddo, N. Linear-time algorithms for linear programming in R3 and related problems. *SIAM J. Comput.* **1983**, *12*, 759–776. [CrossRef]
29. Brass, P.; Knauer, C.; Na, H.; Shin, C.; Vigneron, A. Computing k-centers on a line. *arXiv* **2009**, arXiv:0902.3282.
30. Sharir, M. A near-linear algorithm for the planar 2-center problem. *Discret. Comput. Geom.* **1997**, *18*, 125–134. [CrossRef]
31. Eppstein, D. Faster construction of planar two-centers. *SODA* **1997**, *97*, 131–138.
32. Agarwal, P.; Sharir, M.; Welzl, E. The discrete 2-center problem. *Discret. Comput. Geom.* **1998**, *20*, 287–305. [CrossRef]
33. Frederickson, G. Parametric search and locating supply centers in trees. In *Workshop on Algorithms and Data Structures*; Springer: Berlin/Heidelberg, Germany, 1991; pp. 299–319.
34. Karmakar, A.; Das, S.; Nandy, S.; Bhattacharya, B. Some variations on constrained minimum enclosing circle problem. *J. Comb. Optim.* **2013**, *25*, 176–190. [CrossRef]
35. Chen, D.; Li, J.; Wang, H. Efficient algorithms for the one-dimensional k-center problem. *Theor. Comput. Sci.* **2015**, *592*, 135–142. [CrossRef]
36. Drezner, Z. On the rectangular p-center problem. *Nav. Res. Logist. (NRL)* **1987**, *34*, 229–234. [CrossRef]
37. Katz, M.J.; Kedem, K.; Segal, M. Discrete rectilinear 2-center problems. *Comput. Geom.* **2000**, *15*, 203–214. [CrossRef]
38. Drezner, Z. On a modified one-center model. *Manag. Sci.* **1981**, *27*, 848–851. [CrossRef]

39. Hansen, P.; Jaumard, B. Cluster analysis and mathematical programming. *Math. Program.* **1997**, *79*, 191–215. [CrossRef]
40. Doddi, S.; Marathe, M.; Ravi, S.; Taylor, D.; Widmayer, P. Approximation algorithms for clustering to minimize the sum of diameters. *Nord. J. Comput.* **2000**, *7*, 185–203.
41. Gibson, M.; Kanade, G.; Krohn, E.; Pirwani, I.A.; Varadarajan, K. On metric clustering to minimize the sum of radii. *Algorithmica* **2010**, *57*, 484–498. [CrossRef]
42. Charikar, M.; Panigrahy, R. Clustering to minimize the sum of cluster diameters. *J. Comput. Syst. Sci.* **2004**, *68*, 417–441. [CrossRef]
43. Behsaz, B.; Salavatipour, M. On minimum sum of radii and diameters clustering. *Algorithmica* **2015**, *73*, 143–165. [CrossRef]
44. Mahajan, M.; Nimbhorkar, P.; Varadarajan, K. The planar k-means problem is NP-hard. *Theor. Comput. Sci.* **2012**, *442*, 13–21. [CrossRef]
45. Shang, Y. Generalized K-Core percolation in networks with community structure. *SIAM J. Appl. Math.* **2020**, *80*, 1272–1289. [CrossRef]
46. Tao, Y.; Ding, L.; Lin, X.; Pei, J. Distance-based representative skyline. In Proceedings of the 2009 IEEE 25th International Conference on Data Engineering, Shanghai, China, 29 March–2 April 2009; pp. 892–903.
47. Cabello, S. Faster Distance-Based Representative Skyline and k-Center Along Pareto Front in the Plane. *arXiv* **2020**, arXiv:2012.15381.
48. Sayın, S. Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. *Math. Program.* **2000**, *87*, 543–560. [CrossRef]
49. Auger, A.; Bader, J.; Brockhoff, D.; Zitzler, E. Investigating and exploiting the bias of the weighted hypervolume to articulate user preferences. In Proceedings of the GECCO 2009, Montreal, QC, Canada, 8–12 July 2009; pp. 563–570.
50. Bringmann, K.; Cabello, S.; Emmerich, M. Maximum Volume Subset Selection for Anchored Boxes. In Proceedings of the 33rd International Symposium on Computational Geometry (SoCG 2017), Brisbane, Australia, 4–7 July 2017; Aronov, B., Katz, M.J., Eds.; Volume 77, pp. 22:1–22:15. [CrossRef]
51. Bringmann, K.; Friedrich, T.; Klitzke, P. Two-dimensional subset selection for hypervolume and epsilon-indicator. In Proceedings of the Annual Conference on Genetic and Evolutionary Computation, Vancouver, BC, Canada, 12–16 June 2014; pp. 589–596.
52. Kuhn, T.; Fonseca, C.; Paquete, L.; Ruzika, S.; Duarte, M.; Figueira, J. Hypervolume subset selection in two dimensions: Formulations and algorithms. *Evol. Comput.* **2016**, *24*, 411–425. [CrossRef]
53. Erkut, E. The discrete p-dispersion problem. *Eur. J. Oper. Res.* **1990**, *46*, 48–60. [CrossRef]
54. Hansen, P.; Moon, I. Dispersing facilities on a network. *Cahiers du GERAD* **1995**.
55. Dupin, N. Polynomial algorithms for p-dispersion problems in a 2d Pareto Front. *arXiv* **2020**, arXiv:2002.11830.
56. Dupin, N.; Nielsen, F.; Talbi, E. k-medoids and p-median clustering are solvable in polynomial time for a 2d Pareto front. *arXiv* **2018**, arXiv:1806.02098.
57. Dupin, N.; Nielsen, F.; Talbi, E. Dynamic Programming heuristic for k-means Clustering among a 2-dimensional Pareto Frontier. In Proceedings of the 7th International Conference on Metaheuristics and Nature Inspired Computing, Marrakech, Morocco, 27–31 October 2018.
58. Grønlund, A.; Larsen, K.; Mathiasen, A.; Nielsen, J.; Schneider, S.; Song, M. Fast exact k-means, k-medians and Bregman divergence clustering in 1d. *arXiv* **2017**, arXiv:1701.07204.
59. Wang, H.; Song, M. Ckmeans. 1d. dp: Optimal k-means clustering in one dimension by dynamic programming. *R J.* **2011**, *3*, 29. [CrossRef]
60. Gmys, J.; Carneiro, T.; Melab, N.; Talbi, E.; Tuyttens, D. A comparative study of high-productivity high-performance programming languages for parallel metaheuristics. *Swarm Evol. Comput.* **2020**, *57*, 100720. [CrossRef]
61. Zio, E.; Bazzo, R. A clustering procedure for reducing the number of representative solutions in the Pareto Front of multiobjective optimization problems. *Eur. J. Oper. Res.* **2011**, *210*, 624–634. [CrossRef]
62. Samorani, M.; Wang, Y.; Lv, Z.; Glover, F. Clustering-driven evolutionary algorithms: An application of path relinking to the quadratic unconstrained binary optimization problem. *J. Heuristics* **2019**, *25*, 629–642. [CrossRef]
63. Pulido, G.; Coello, C. Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer. In Proceedings of the Genetic and Evolutionary Computation Conference, Seattle, WA, USA, 26–30 June 2004; pp. 225–237.