Clustering to the Fewest Clusters Under Intra-Cluster Dissimilarity Constraints

Jennie Andersen, Brice Chardin, Mohamed Tribak

Clustering to the Fewest Clusters Under Intra-Cluster Dissimilarity Constraints

Jennie Andersen
University of Lyon, INSA Lyon, CNRS UMR 5205
jennie.andersen@liris.cnrs.fr

Brice Chardin
ISAE-ENSMA, LIAS
brice.chardin@ensma.fr

Mohamed Tribak
SRD, LIAS
mohamed.tribak@srd-energies.fr

***Abstract*—This paper introduces the equiwide clustering problem, where valid partitions must satisfy intra-cluster dissimilarity constraints. Unlike most existing clustering algorithms, equiwide clustering relies neither on density nor on a predefined number of expected classes, but on a dissimilarity threshold. Its main goal is to ensure an upper bound on the error induced by ultimately replacing any object with its cluster representative. Under this constraint, we then primarily focus on minimizing the number of clusters, along with potential sub-objectives.**

We argue that equiwide clustering is a sound clustering problem, and discuss its relationship with other optimization problems, existing and novel implementations as well as approximation strategies. We review and evaluate suitable clustering algorithms to identify trade-offs between the various practical solutions for this clustering problem.

I. Introduction

When the expected number of classes is unknown, there exists no universal method to determine the number of clusters [1]. To circumvent this problem, several algorithms instead rely on density to detect groups of elements. However, density-based clustering can lead to elongated clusters, which allows for relatively dissimilar elements to belong to the same cluster.

In this work, we are concerned with clustering problems for which the number of clusters is to be determined, with strong guaranties on the dissimilarity of elements belonging to the same cluster. We therefore introduce the equiwide clustering problem, where valid partitions must satisfy intra-cluster dissimilarity constraints. Unlike most existing clustering algorithms, equiwide clustering relies neither on density nor on a predefined number of expected classes, but on a dissimilarity threshold. Its main goal is to ensure an upper bound on the error induced by ultimately replacing any object with its cluster representative. Under this constraint, we then primarily focus on minimizing the number of clusters, and discuss potential sub-objectives. Equiwide clustering aims at solving problems for which the dissimilarity measure can be reasoned upon by domain experts.

Industrial use case: SRD is an electricity distribution network operator that manages its network on a regional level over a surface of 7000 km^2, with 12,323 km of network lines, 16 step-down substations and more than 8000 public distribution substations. To optimize the topology of its network, SRD models the behavior of its step-down substations, that connect the national transmission network (with voltages ranging from 63 kV to 400 kV) to its local distribution network (with a voltage of 20 kV). Since there are 180 substations output power lines, modeling each of them individually is impractical. A first step consists in grouping them into categories to then define a common model.

Yet, what categories exist is unknown. In particular, their number is to be determined. Network operators at SRD are only able to specify a similarity measure and a threshold under which two substation outputs could be considered as belonging to the same category. The goal is, hopefully, to identify the fewest number of categories in order to simplify subsequent modeling work in the network optimization process.

Other practical applications of equiwide clustering are taken from related optimization problems (presented in Section II), such as scheduling or radio frequency assignment [2]. In this paper, we discuss this kind of clustering problem and solutions from the state of the art.

Paper organization: In the remainder of this section, we provide preliminary definitions and formalize the equiwide clustering problem. In Section II, we discuss its relationship with other optimization problems, along with other approaches from the state of the art. In Section III, we consider various implementation strategies and approximations. We then provide an experimental evaluation of relevant algorithms in Section IV.

A. Preliminary definitions

Definition 1 (Population)**.** *Let $X = \{x_1, ..., x_n\}$ be a finite set of n elements to be partitioned.*

Definition 2 (Partition)**.** *Let $P = \{C_1, ..., C_k\}$ be a partition of X:*

$$\bigcup_{C_i \in P} C_i = X \quad \text{(cover)}$$
$$\forall C_i \in P, C_i \neq \emptyset \quad \text{(non-emptiness)}$$
$$\forall C_i, C_j \in P, i \neq j \Rightarrow C_i \cap C_j = \emptyset \quad \text{(pairwise disjunction)}$$

Definition 3 (Dissimilarity)**.** *Let d be a dissimilarity on pairs of elements of X, $\forall a, b \in X$:*

$$d(a, a) = 0$$
$$d(a, b) = d(b, a) \quad \text{(symmetry)}$$
$$d(a, b) \geq 0 \quad \text{(positivity)}$$

We consider two concepts of wideness to characterize the homogeneity of a subset of elements of X, its diameter and its radius.

Definition 4 (Diameter)**.** *The diameter of a subset A of X is the maximum dissimilarity between pairs of elements of A.*

$$\mathrm{D}(A) = \max_{a,b \in A} d(a,b)$$

Definition 5 (Radius)**.** *The radius of a subset A of X, first defined as a measure of homogeneity of a cluster [3], is the minimum eccentricity of any element of A.*

$$\mathrm{R}(A) = \min_{a \in A} \max_{b \in A} d(a,b)$$

Definition 6 (Homogeneity)**.** *A subset A of X is called homogeneous (specifically diameter-homogeneous or radius-homogeneous) under a provided threshold T if its wideness (diameter or radius) is at most equal to this threshold. In the remainder of this paper, a diameter-specific threshold will be noted D_{max} and a radius-specific threshold will be noted R_{max}.*

A partition P of X is called homogeneous if each set in P is homogeneous.

In the next two definitions, we are concerned with the identification of a representative element for a subset A of X. This element, called the *center* of A, has the smallest maximal dissimilarity with every element of A.

Definition 7 (Center, diameter – normed vector space)**.** *If X is a subset of a normed vector space, the center is the midpoint of the two furthest elements in A, i.e., the pair of elements whose dissimilarity is equal to the diameter. This center is not necessarily an element of A itself.*

$$\text{D-center}(A) = \frac{a+b}{2}, \quad (a,b) = \operatorname*{argmax}_{(i,j) \in A \times A} d(i,j)$$

Definition 8 (Center, radius)**.** *The central element of A is the element whose greatest dissimilarity with every element of A is minimal. This center is not necessarily the medoid, as a medoid minimizes the average dissimilarity.*

$$\text{R-center}(A) = \operatorname*{argmin}_{a \in A} \max_{b \in A} d(a,b)$$

B. Problem statement

Definition 9 (Equiwide clustering)**.** *For a given population X, a dissimilarity d and a threshold T, we are concerned with the identification of a partition P of X so that:*

- *P is homogeneous under T,*
- *the number of clusters $|P|$ is minimal, i.e., there exists no partition P' homogeneous under T where $|P'| < |P|$.*

When we are concerned with the identification of a representative element for each cluster with an upper-bound on its dissimilarity with every element of the cluster, a diameter constraint will not provide a tight upper bound if a center (diameter) cannot be computed. This problem notably occurs when the elements to be partitioned are not defined within a normed vector space. In that case, a radius constraint offers tighter dissimilarity guaranties between a representative element—the center (radius) of each cluster—and every other element of the cluster. While conceptually similar, these two types of wideness constraints (diameter or radius) introduce significant variations on how the problem can be resolved efficiently.

Sub-objectives: This clustering problem may have multiple solutions with the same number of clusters, and sub-objectives can be added to discriminate between them. These sub-objectives may vary depending on the data to be partitioned. In this paper, we consider three sub-objectives.

Minimizing the maximum width: This subgoal aims at providing tighter upper bounds on the intra-cluster dissimilarity constraint considered—either the diameter or the radius.

$$\text{minimize} \max_{C_i \in P} \mathrm{D}(C_i) \quad \text{or} \quad \text{minimize} \max_{C_i \in P} \mathrm{R}(C_i)$$

Minimizing the within-cluster sum of dissimilarities: This global homogeneity measure of a partition has been considered as a sub-objective in related work [4].

$$\text{minimize} \sum_{C_i \in P} \sum_{a,b \in C_i} d(a,b)$$

Maximizing the variance of cluster sizes: In the context of SRD described in Section I, experts prefer to obtain few large clusters and some outliers, rather than clusters of average size. The variance on cluster sizes captures this criterion and has therefore been considered in this study.

$$\text{maximize} \sum_{C_i \in P} |C_i|^2$$

II. Related Work

A. Equivalent problems

The equiwide clustering problem—as defined in definition 9, without sub-objectives—has been considered in graph theory. Some theoretical results therefore apply, as well as algorithms to compute partitions with a minimal number of clusters. Yet, since sub-objectives are absent from these formulations, resulting partitions may not be entirely satisfactory.

Let $G = (X, E)$ be a graph where $E = \{\{x_i, x_j\} \mid d(x_i, x_j) \leq T\}$, namely there is an edge between vertices if and only if the dissimilarity between them is at most equal to the provided threshold, i.e., adjacent vertices are compatible.

Minimum clique cover: The diameter-based clustering problem is equivalent to the minimum (vertex) clique cover problem in graph theory. Each resulting clique represents a cluster, within which each pair of element is compatible, i.e., their dissimilarity is at most equal to the diameter. This problem is NP-hard [5].

Graph coloring: Let G' be the complement graph of G, i.e., vertices of G' are adjacent if and only if they are incompatible. The minimum clique cover on graph G is equivalent to coloring vertices of G' using a minimal number of colors. The resulting colors represent clusters: vertices of the same color belong to the same cluster.

Minimal cardinality dominating set: A dominating set of G is a subset D of X so that every vertex of $X \setminus D$ is adjacent to at least one vertex of D. The radius-based clustering problem is equivalent to the minimal cardinality dominating set problem in graph theory. Each dominating element is the center (radius) of a cluster. This problem is NP-hard [6].

B. Related clustering algorithms

Only a few existing clustering techniques meet, at least partially, the requirements of equiwide clustering. For instance, the well-known k-means algorithm requires the number of clusters to be provided and cannot respect a dissimilarity-based constraint. A constraint on a maximal diameter can be added by means of instance-level cannot-link constraints (COP-Kmeans) [7], but the number of clusters is still required. While density-based clustering algorithms can be used to identify the number of clusters, they do not enforce wideness constraints on the resulting partition, and are therefore not considered in this paper.

1) Hierarchical clustering: Hierarchical clustering techniques build a hierarchy of clusters that can later be used to create a partition. A cut-off criterion can be defined, either as a number of clusters or as a dissimilarity threshold. The complete-link agglomerative hierarchical clustering is a fast algorithm on which a maximum diameter constraint can be set. The complete-link criterion merges two clusters A and B based on the maximum dissimilarity between pairs of elements of each cluster.

$$\text{distance}\,(A, B) = \max_{a \in A, b \in B} d\,(a, b)$$

If the diameter threshold is applied on this distance, the maximum diameter criterion is respected. While the complete-link agglomerative hierarchical clustering with a cut-off distance is a valid equiwide clustering method with respect to its main constraint (wideness – diameter only), it does not provide any optimality guarantee. In particular, this algorithm does not attempt to minimize the number of clusters. It is also not applicable to a radius constraint, apart from considering the pairwise dissimilarity as a loose upper bound.

2) Graph coloring: Graph coloring algorithms that minimize the number of colors can be used to cluster elements with a maximum diameter on the associated graph (see Section II-A). Hansen and Delattre [8] proposed an algorithm, called CLUSTERGRAPH, that also minimizes the maximum diameter of clusters. This algorithm relies on an external optimal graph coloring algorithm. While this algorithm provides a partition that is optimal on both the main objective—minimizing the number of clusters—and the maximum diameter sub-objective, it is also not usable with a radius constraint.

3) Constraint programming: A constraint programming algorithm has been proposed by Dao et al. [4] for constrained clustering. Among the existing constraints, it is possible to specify a maximum diameter. A few objective functions have been defined, including minimizing the maximum diameter or minimizing the within-cluster sum of dissimilarities. With their algorithm, it is possible to find an optimal solution for these objectives or to stop at the first solution satisfying the maximum diameter constraint. While their proposition does not directly minimize the number of clusters—this value is an input of the algorithm—a partition with a minimum number of clusters can be identified by iterating over the number of clusters until a valid solution is found.

In order to improve the run time of the algorithm, the authors suggest to reorder elements according to a Furthest-Point-First (FPF) heuristic: the first element is the one furthest from all the other elements, then the following elements are ordered by decreasing dissimilarity with all previously identified furthest points. This heuristic is used to guide the search strategy, but also to provide a lower and upper bound on the optimal diameter during the optimization process.

4) Cannot-link constraints: An intra-cluster dissimilarity constraint is a cluster-level constraint, but it can also be expressed as an instance-level cannot-link constraint. Two elements affected by a cannot-link constraint cannot belong to the same cluster. Consequently, a maximum diameter constraint can be translated into multiple cannot-link constraints between each pair of elements whose dissimilarity is greater than D_{max}. These instance-level constraints have been considered to augment existing clustering algorithms such as DBSCAN [9] and k-means [7].

III. Equiwide clustering

A. Constraint programming and linear programming formulations

The problem of minimizing the number of clusters with respect to a maximum radius or diameter can be formulated as a constrained optimization problem. This formulation allows for solvers to be used as a baseline during the evaluation of the various implementations.

Radius constraint: Let r be an $n \times n$ Boolean matrix. The value r_{ij} represents whether element i belongs to the cluster centered on element j. The value r_{jj} indicates if element j is the *center* of a cluster.

$$\text{minimize} \quad \sum_j r_{jj} \tag{1a}$$

$$\text{subject to} \quad \sum_j r_{ij} = 1 \qquad i \in \{1, \ldots, n\} \tag{1b}$$

$$d(i,j) \times r_{ij} \leq R_{max} \qquad i, j \in \{1, \ldots, n\} \tag{1c}$$

$$r_{jj} = \max_i (r_{ij}) \qquad j \in \{1, \ldots, n\} \tag{1d}$$

$$r_{ij} \in \{0, 1\} \qquad i, j \in \{1, \ldots, n\} \tag{1e}$$

The first constraint (1b) asserts that every element is represented exactly once. The second constraint (1c) asserts that, if j represents i, then the dissimilarity between i and j is at most equal to the maximum radius. Constraint (1d) asserts that, if i is represented by j, then j is a center.

The objective function (1a) is the number of clusters, to be minimized. It is also possible to add a sub-objective minimizing the sum of dissimilarities between elements and

their center. If the dissimilarities are not all zero, then the objective function becomes:

$$\sum_j r_{jj} + \frac{\sum_{i,j} d(i,j) \times r_{ij}}{\sum_{i,j} d(i,j)}$$

The second half of this objective function belongs to $[0, 1)$, ensuring that the optimal number of clusters will not be influenced.

Diameter constraint: The maximum diameter problem can also be formulated as a constraint programming problem. Let l_i be the cluster label of element i and k the number of clusters.

$$\begin{aligned}
&\text{minimize} && k && \text{(2a)}\\
&\text{subject to} && l_i \neq l_j \quad i,j \in \{1,\dots,n\} \mid d(i,j) > D_{max} && \text{(2b)}\\
& && k = \max_i(l_i) && \text{(2c)}\\
& && l_i \in \{0,\dots,n-1\} \qquad i \in \{1,\dots,n\} && \text{(2d)}
\end{aligned}$$

It is possible to reduce the search space by fixing labels for an independent set of elements, i.e., a set of elements that are pairwise incompatible. Since finding the maximum independent set in a graph is NP-hard, this process has to rely on heuristics to be efficient—similar to the FPF heuristic in [4].

In order to transform these constrained optimization problem into integer linear programming problems, maximum equality constraints (1d) and (2c) should be replaced by multiple inequalities. For instance in the radius formulation, the expression $r_{jj} = \max_i(r_{ij})$ can be translated into $\forall i : r_{jj} \geq r_{ij}$.

B. Decomposition into subproblems

Instead of considering the problem as a whole, equiwide clustering can be split into three processing steps:

1) Enumerate all maximal homogeneous sets under the given threshold.
2) Compute a minimal set cover of the population by homogeneous sets.
3) Assign each element to a unique set of the cover.

Enumerating homogeneous sets: The first step of equiwide clustering is to find all maximal homogeneous sets, that is, homogeneous sets that are not proper subsets of other homogeneous sets. With a radius constraint, this task is straightforward: each element x_i is successively considered as a center to compute the associated homogeneous set S_i. This set contains all elements that are dissimilar from x_i by at most R_{max}, i.e., $S_i = \{e \mid d(x_i, e) \leq R_{max}\}$. This process guarantees that all maximal homogeneous sets are found, but potentially non-maximal homogeneous sets are also computed. There are at most as many homogeneous sets as the number n of elements to be partitioned.

Proof that $\{S_1, \dots, S_n\}$ contains all maximal homogeneous sets. Let S be a maximal radius-homogeneous set. We show that $S \in \{S_1, \dots, S_n\}$. Let x_i be the center of S. The element x_i belongs to X, so there exists $S_i \in \{S_1, \dots, S_n\}$ such that $S_i = \{e \mid d(x_i, e) \leq R_{max}\}$. We will show that $S = S_i$. Let $a \in S$, as S is homogeneous, we have $d(x_i, a) \leq R_{max}$. According to the definition of S_i, we then have $a \in S_i$. This proves that $S \subseteq S_i$. As S is maximal and S_i is homogeneous, we conclude that $S = S_i$. □

For a diameter constraint, enumerating all maximal homogeneous sets is much more difficult. They are the maximal cliques in the associated graph $G = (X, E)$, where $E = \{\{x_i, x_j\} \mid d(x_i, x_j) \leq D_{max}\}$. A set of vertices $C \subseteq X$ is a clique if each vertex of C is adjacent to every other vertex of C, i.e., there is an edge between each pair of element of C. A clique is maximal if it does not have any strict superset which is also a clique.

Proposition 1. *Maximal diameter-homogeneous sets are the maximal cliques of G.*

Proof. Let us show that $A \subseteq X$ is a diameter-homogeneous set if and only if A is a clique of G. Let $A \subseteq X$ be a diameter-homogeneous set. Then, for all pairs of elements (x_i, x_j) of A, we have $d(x_i, x_j) \leq D_{max}$ because A is homogeneous. Hence, $\{x_i, x_j\}$ is an edge of E and x_i and x_j are adjacent. Consequently, each pair of elements of A are adjacent, so A is a clique. Now, if A is a clique, then all pairs of elements of A are adjacent, and so they are dissimilar by at most D_{max}, hence A is diameter-homogeneous. □

Enumerating all maximum cliques of a graph is a well-studied problem with many available algorithms and implementations, e.g. [10], [11]. This problem is NP-hard and can generate up to $3^{n/3}$ maximal diameter-homogeneous sets [12].

Computing a minimal set cover: From the set of homogeneous sets, the second step consists in identifying a minimal cover of the population. This determines the number of clusters and, for the most part, the clusters themselves. The minimal set cover is a well-known NP-hard problem [5] defined as follows. Let S be a set of subsets of X, i.e., $S \subseteq \mathbb{P}(X)$. A minimal set cover of X by S is a set $T \subseteq S$ such that:

$$\bigcup_{E \in T} E = X \qquad \text{(cover)}$$

$$\nexists\, T' \subseteq S,\ |T'| < |T| \wedge \bigcup_{E \in T'} E = X \qquad \text{(minimality)}$$

The resulting cover T is not necessarily a valid partition since sets of the cover are not required to be pairwise disjoint. Yet, clusters will be subsets of sets in this cover. The minimality condition guarantees that the number of clusters is minimal. Various methods exist to solve this problem, the result can either be a single minimal solution or an enumeration of all minimal solutions. Enumerating all solutions can be beneficial—or event required—in order to optimize the considered sub-objective, by then selecting the most appropriate cover in the enumeration.

Assigning each element to a unique cluster: Some elements can belong to several sets in the cover computed by the previous step. These are referred to as undecided elements. We consider approximate assignement strategies for each

sub-objective. To minimize both the within cluster sum of dissimilarities and the maximum diameter, undecided elements are assigned to the cluster with the closest center. This center is computed after removal of all undecided elements. To maximize the cluster sizes variance, a greedy algorithm is implemented to assign undecided elements to the largest clusters. In both cases, the homogeneity of resulting clusters is trivially guaranteed for a diameter constraint. However, with a maximum radius criterion, the center (radius) of each cluster should not be removed when assigning undecided elements.

C. Implementation

The current implementation of equiwide clustering only computes comprehensive enumerations of maximal homogeneous sets. This can be prohibitive with a diameter constraint due to the the large number ($3^{n/3}$) of maximal cliques in the worst case. However, some approximation strategies exist and could be considered [13].

The second step computes a minimal set cover, which determines the number of clusters. We have implemented three methods to solve this problem, depending on how optimal the result should be. The first method (EQW-Exhaustive) consists in enumerating all minimal solutions [14]. The second method (EQW-LP) uses integer linear programming to find a single minimal solution [15]. In this second implementation, variables x_i denote whether set S_i of S should be included in the cover of X. The objective is to minimize the number of sets in the cover while covering every element by at least one set.

$$\begin{aligned} \text{minimize} \quad & \sum_{i|S_i \in S} x_i \\ \text{subject to} \quad & \sum_{i|S_i \in S \wedge e \in S_i} x_i \geq 1 \qquad e \in X \\ & x_i \in \{0,1\} \qquad\qquad\quad i \mid S_i \in S \end{aligned}$$

The third method (EQW-Greedy) is a greedy algorithm which selects the largest clusters until all elements are covered. This algorithm finds a set cover in polynomial time, and is H_n-competitive with $H_n = \sum_{k=1}^{n} \frac{1}{k} \leq ln(n) + 1$ [16].

These three possibilities to compute the set cover impact both the optimality of the solution on the primary or secondary objectives, and the execution time.

IV. Experimental Evaluation

In order to compare the algorithms described in this paper, an experimental evaluation has been conducted. Considered algorithms from the state of the art are listed below, with their objectives summarized in Table I.

- Complete-link hierarchical agglomerative clustering (HAC), using the scikit-learn implementation [17].
- Exact graph coloring, using the DSATUR-based implementation provided by Mehrotra and Trick [18].
- Graph coloring with minimization of the maximum diameter of clusters [8] (denoted CG, for CLUSTERGRAPH), implemented in Python and relying on DSATUR.
- Constraint programming for constrained clustering [4] (CP4CC), using the reference implementation provided by the authors.

We compared these four algorithms to the propositions introduced in this paper, namely:

- Linear programming (LP), implemented with the Coin-or branch and cut solver.
- Constraint programming (CP), implemented with Google's OR-Tools CP-SAT solver.
- All variants of equiwide clustering (EQW denotes this family of algorithms in general, while EQW-LP and EQW-Greedy identify respectively the linear programming and the greedy implementations for the computation of the minimal set cover).

These algorithms are tested on 11 datasets from the UCI Machine Learning Repository [19]. The maximum diameter threshold is taken from [4], where this diameter was computed by minimizing the maximum diameter of the clusters when the number of clusters is equal to the number of real classes[1]. In these experiments, thresholds are rounded up to avoid round-off error. The considered dissimilarity is the Euclidean distance. Half (five) of these datasets were normalized by their authors, meaning that the ranges of their attributes are comparable, either inherently or as a post-processing step. The other half (six) were not normalized, the weight of their attributes can therefore vary significantly—by a factor of up to 140,000 for WDBC—when computing the Euclidean distance. Table II provides a summary of the datasets used in this experimental evaluation. When the dataset is not normalized, an indication of the disparity between attributes is provided as the ratio between the largest and smallest ranges.

Algorithms are also tested on 12+1 datasets provided by SRD. Each element in these datasets is a time series of electric power passing through one of the 145 busbar output of step-down substations[2]. These time series are normalized with respect to the total power subscription of connected clients. Each of the 12 datasets corresponds to a month of 2017 with a sampling period of ten minutes. A 13th dataset is constructed as the union of the previous 12, therefore containing 145×12 time series[3]. Each dataset is converted into a dissimilarity matrix using Dynamic Time Warping (DTW) with a window of 4, which serves as the input of clustering algorithms. The computation time of this dissimilarity matrix is not taken into account in these experiments. The maximum diameter threshold is 0.1, which has been determined by network operators at SRD as an acceptable margin.

Each algorithm is executed on each dataset with four randomized ordering of elements. For algorithms with a radius constraint, the maximum radius is defined as half the value of the maximum diameter. Experiments are performed on an

[1]For image segmentation, the optimal diameter had been computed on only 2000 of the 2100 elements, resulting in a lower threshold.

[2]Out of the 180 available, 35 outputs were not included in this study due to maintenance operations over the period and other practical considerations.

[3]Series of different length were truncated during the computation of the DTW dissimilarity to accommodate for months with fewer than 31 days.

TABLE I
EVALUATED ALGORITHMS

Algorithm	Minimal #clusters	Sub-objective	Optimal	Wideness constraint
Complete-link HCA [17]	no	–	–	diameter
DSATUR [18]	yes	–	–	diameter
CLUSTERGRAPH [8]	yes	maximum diameter	yes	diameter
CP4CC [4]	iteratively	maximum diameter WCSD	yes	diameter
Equiwide clustering	yes	variance WCSD	no	diameter radius
Integer linear programming	yes	–	–	diameter radius
Constraint programming	yes	–	–	diameter radius

TABLE II
EXPERIMENTAL EVALUATION DATASETS

Name	Origin	#Elements	#Dimensions	#Classes	Normalized (disparity)	Dissimilarity	D_{max}
Iris	UCI	150	4	3	no (2.5)	ED	2.59
Wine	UCI	178	13	3	no (2.6×10^3)	ED	458.14
Glass Identification	UCI	214	9	7	no (4.7×10^2)	ED	4.98
Ionosphere	UCI	351	34	2	yes	ED	8.7
User Knowledge	UCI	403	5	4	yes	ED	1.18
WDBC	UCI	569	30	2	no (1.4×10^5)	ED	2377.97
Synthetic Control	UCI	600	60	6	yes	ED	109.37
Vehicle	UCI	846	18	4	no (7.0×10^1)	ED	264.84
Yeast	UCI	1484	8	10	yes	ED	0.68
Image Segmentation	UCI	2100	19	7	no (6.2×10^3)	ED	436.5
Waveform	UCI	5000	40	3	yes	ED	15.7
Monthly elec. power	SRD	145 ($\times$ 12)	4032–4464	unknown	yes	DTW	0.1
Yearly elec. power	SRD	1740	4032–4464	unknown	yes	DTW	0.1

ED: Euclidean Distance
DTW: Dynamic Time Warping with a window of 4

2.40 GHz Intel Xeon CPU E5-2630 processor, with 32 GB of RAM, running Ubuntu 18.04. The Python interpreter is CPython 3.8.0. Each test has a maximum run time of ten minutes. None of the algorithms is parallelized.

Number of clusters: We compare separately algorithms allowing a diameter constraint and algorithms allowing a radius constraint, but the same criteria of quality apply. First, every clustering has to satisfy the provided wideness constraint, which is the case in every experiment. Then, the main quality criterion is the number of resulting clusters, as the objective is to minimize this value. Experimental results are displayed in Table III (columns Radius and Diameter). In this table, the Opt. column refers to all algorithms which gave the same optimal number of clusters. When two values are provided, they represent the minimum and the maximum number of clusters among the different orderings. As expected by their design, every algorithm except for HAC and the greedy variant of EQW provides the optimal number of cluster.

Since the specified diameter threshold is tight for the UCI datasets, we also experimented with a threshold set to 1.2 times the optimal diameter, in order to evaluate the impact of having relaxed constraints and potentially a larger solution space. Results are presented in Table III (column 1.2 $\times$ Diameter). With these new thresholds, results from both approximate algorithms become closer to optimal values.

Overall, the greedy version of EQW usually provides slightly fewer clusters than HAC, with some guarantees related to its H_n-competitiveness. For instance, with the ionosphere dataset, the greedy version of EQW guarantees to find a solution with at most $\lfloor 2H_2 \rfloor = 3$ clusters, while HAC provides a solution with 4 clusters.

Execution time: With diameter constraints, the constraint programming implementation significantly outperforms the linear programming implementation. On the opposite, with radius constraints, the linear programming implementation outperforms the constraint programming one. Only results for the best of these two baseline implementations are presented.

On all datasets except for iris, wine and monthly electric power, the variant of EQW that performs an exhaustive enumeration of minimal set covers fails to provide a solution within 10 minutes. Its results are therefore omitted from this analysis. With diameter constraints, the predominance of the first step—the enumeration of homogeneous sets—in the execution time causes all variants of EQW to be within a few

TABLE III
NUMBER OF CLUSTERS

Dataset	Radius		Diameter			1.2 × Diameter		
	EQW-Greedy	Opt.	EQW-Gr.	HAC	Opt.	EQW-Gr.	HAC	Opt.
Iris	**4**	**4**	4	4	**3**	4	4	**3**
Wine	**4**–5	**4**	4	4	**3**	**3**–4	4	**3**
Glass Identification	**13**–14	**13**	8–9	10	**7**	**6**	7	**6**
Ionosphere	**28**–29	**28**	**2**–3	4	**2**	**1**	**1**	**1**
User knowledge	9–11	**8**	–	7	**4**	**2**–3	**2**	**2**
WDBC	**3**	**3**	**2**	3	**2**	**2**	**2**	**2**
Synthetic Control	16–17	**14**	–	9	**6**	**4**–6	5	**4**
Vehicle	6–7	**5**	5	6	**4**	**4**	5	**4**
Yeast	21	**18**	–	14–15	**10**	8–9	9	**7**
Image Segmentation	13	**12**	**8**	**8**	**8**	**5**	**5**	**5**
Waveform	168–172	**149**	–	6	**3**	–	3	**2**
Elec. power (July)	**13**–14	**13**	9	10	**8**			
Yearly elec. power	**16**	**16**	–	30	**19**			

Execution times exceeding the 10 minutes limit are denoted by –

percent of each other. Their execution times are consequently combined under the EQW label. With radius constraints, the second step—the computation of a minimal set cover—becomes predominant and variants of EQW are then split between the EQW-Greedy and the EQW-LP labels.

The execution time on the monthly electric power datasets being similar, only a representative month, July, is reported in the results. Execution times are reported in Table IV for diameter constraints, and Table V for radius constraints.

HAC is by far the fastest algorithm, but it rarely reaches the minimal number of clusters. The next fastest algorithm is DSATUR, followed by CLUSTERGRAPH. The former is always faster than the latter since CLUSTERGRAPH makes multiple calls—at least two—to DSATUR.

In the reference implementation of CP4CC, the FPF heuristic can not be used with a dissimilarity matrix as the input. Consequently, on the datasets provided by SRD, this heuristic is not used and CP4CC exceeds the time limit of 10 minutes on 26 of the 48 experiments performed with the monthly electric power datasets, despite them being relatively simple compared with other datasets provided by UCI. This behavior highlights the influence of heuristics to solve the equiwide clustering problem. The FPF reordering also allows CP4CC to be independent on the innate ordering of the input data. Otherwise, exact algorithms—DSATUR, CP and CP4CC without the FPF heuristic—are all dependent on this ordering, as most notably seen on the SRD dataset, where there is at least a tenfold difference between the fastest and slowest run.

EQW does not follow the performance pattern of other algorithms, for which the execution time is mostly related to the number of elements in the dataset. This is explained by the potentially large number of homogeneous sets that have to be computed. For instance, EQW could not complete on User Knowledge which only contains 403 elements, and for which there can theoretically be up to 1.2×10^{64} maximal homogeneous sets. In fact, after ten minutes, EQW already identified 3.9×10^{7} maximal homogeneous sets. The main issue in addressing this problem is that it is not possible to efficiently predict this behavior from the dissimilarity matrix. This limitation on how EQW computes homogeneous sets with a diameter constraint severely limits its ability to provide exact solutions in a reasonable time compared to approaches from the state of the art. It then seems only appropriate to use EQW if this step can be approximated upon reaching an excessive number of homogeneous sets.

The limitation on homogeneous sets is not present with radius constraints. EQW then becomes a competitive solution, the baseline linear programming and constraint programming implementations being comparatively much slower. The greedy variant of EQW does not provide a minimal number of clusters, but is applicable on every considered dataset since its overall complexity is polynomial. Alternatively, potentially better approximate solutions can be provided with a slight modification of the linear programming implementation of EQW. For example with the waveform dataset, EQW-LP fails to provide an optimal result within 10 minutes. However, the underlying solver allows for a time limit to be set and, with a one minute limit, finds an approximate solution with 150 clusters—instead of the optimal value of 149—that still improves upon the 168 clusters solution identified by the greedy variant.

V. CONCLUSION

In this work, we provide an homogeneous vision of two problems from graph theory under a common clustering perspective. This leads to the definition of the equiwide clustering problem. The underlying dissimilarity measure becomes a core component that must be intelligible, as a threshold has to be provided according to the application needs. Given the importance of both the dissimilarity and the threshold, a semi-supervised approach could be considered to assist the user in this task, for instance using metric learning [20].

For distance-based intra-cluster constraints, graph coloring-based algorithms appear to be the most promising solutions

TABLE IV
EXECUTION TIME (IN SECONDS) FOR ALGORITHMS WITH A DIAMETER CONSTRAINT

Dataset	HAC	DSATUR	CG	CP4CC	CP	EQW
Iris	< 0.01	0.07 ± 0.0	0.14 ± 0.0	0.11 ± 0.0	0.47 ± 0.0	0.07 ± 0.1
Wine	< 0.01	0.08 ± 0.0	0.15 ± 0.0	0.20 ± 0.0	0.44 ± 0.0	0.10 ± 0.0
Glass Identification	< 0.01	0.09 ± 0.0	0.18 ± 0.0	0.52 ± 0.0	0.26 ± 0.0	0.12 ± 0.1
Ionosphere	< 0.01	0.08 ± 0.0	0.10 ± 0.0	0.74 ± 0.0	0.02 ± 0.0	0.05 ± 0.0
User Knowledge	< 0.01	0.13 ± 0.0	0.22 ± 0.0	0.85 ± 0.0	0.22 ± 0.0	–
WDBC	0.01 ± 0.0	0.29 ± 0.0	0.42 ± 0.0	1.54 ± 0.1	0.55 ± 0.1	0.14 ± 0.0
Synthetic Control	0.01 ± 0.0	0.62 ± 0.1	1.07 ± 0.0	8.29 ± 0.0	11.44 ± 6.7	–
Vehicle	0.02 ± 0.0	1.08 ± 0.0	1.76 ± 0.0	5.49 ± 0.1	23.50 ± 0.7	45.46 ± 3.7
Yeast	0.06 ± 0.0	2.29 ± 0.0	4.83 ± 3.9	29.87 ± 0.1	10.11 ± 1.1	–
Image Segmentation	0.13 ± 0.0	4.64 ± 0.1	9.17 ± 1.1	70.68 ± 1.5	4.68 ± 0.6	1.31 ± 0.1
Waveform	0.91 ± 0.0	32.68 ± 1.7	41.38 ± 2.6	239.98 ± 6.8	483.65 ± 23	–
Elec. power (July)	< 0.01	0.03 ± 0.0	0.07 ± 0.0	$11.58 \pm$ TL	0.14 ± 0.0	0.04 ± 0.0
Yearly elec. power	0.10 ± 0.0	$3.87 \pm$ TL	–	–	$55.60 \pm$ TL	–

$\pm$ TL indicates that the 10 minutes limit has been exceeded in at least one of the four runs

TABLE V
EXECUTION TIME FOR ALGORITHMS WITH A RADIUS CONSTRAINT

Dataset	EQW-Greedy	EQW-LP	LP
Iris	5.43 ± 0.56 ms	45.51 ± 3.25 ms	0.78 ± 0.04 s
Wine	7.97 ± 0.55 ms	50.30 ± 1.57 ms	2.28 ± 1.02 s
Glass Identification	14.61 ± 0.82 ms	67.35 ± 5.07 ms	1.77 ± 0.10 s
Ionosphere	55.78 ± 1.65 ms	0.17 ± 0.02 s	6.60 ± 1.09 s
User Knowledge	46.74 ± 0.96 ms	0.24 ± 0.01 s	18.12 ± 10.0 s
WDBC	0.21 ± 0.03 s	0.34 ± 0.02 s	157.47 ± 111 s
Synthetic Control	74.58 ± 5.00 ms	0.23 ± 0.01 s	26.28 ± 11.2 s
Vehicle	0.37 ± 0.01 s	0.80 ± 0.01 s	78.36 ± 130 s
Yeast	2.43 ± 0.04 s	7.34 ± 0.13 s	–
Image Segmentation	11.27 ± 0.08 s	16.26 ± 0.27 s	–
Waveform	7.69 ± 0.21 s	–	–
Elec. power (July)	8.05 ± 0.44 ms	55.76 ± 3.58 ms	0.86 ± 0.01 s
Yearly elec. power	6.64 ± 0.10 s	11.74 ± 0.09 s	–

from the state of the art. For larger datasets, the complete-link agglomerative hierarchical clustering provides a valid yet non-optimal solution in a much shorter time. EQW, in both its exact and approximate variants, is not competitive in this setting.

The extent to which sub-objectives can be integrated into minimal cardinality dominating set computation algorithms remains to be determined in order to assess their applicability as clustering algorithms for radius constraints. Yet, EQW fills this gap and offers exact or approximate solutions with acceptable performance when a representative element for each cluster has to be selected from the original dataset.

ACKNOWLEDGMENT

This work was supported by Région Nouvelle-Aquitaine and the aLIENOR ANR LabCom (ANR-19-LCV2-0006).

REFERENCES

[1] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, no. 2, pp. 159–179, 1985.

[2] D. J. Johnson and M. A. Trick, *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Workshop, October 11-13, 1993*. American Mathematical Society, 1996.

[3] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Mathematical programming*, vol. 79, no. 1, pp. 191–215, 1997.

[4] T.-B.-H. Dao, K.-C. Duong, and C. Vrain, "Constrained clustering by constraint programming," *Artificial Intelligence*, vol. 244, pp. 70–94, 2017.

[5] R. M. Karp, *Reducibility among Combinatorial Problems*. Springer, 1972, pp. 85–103.

[6] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[7] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained K-Means Clustering with Background Knowledge," in *Proceedings of the 18th International Conference on Machine Learning*, 2001, p. 577–584.

[8] P. Hansen and M. Delattre, "Complete-Link Cluster Analysis by Graph Coloring," *Journal of the American Statistical Association*, vol. 73, no. 362, pp. 397–403, 1978.

[9] C. Ruiz, M. Spiliopoulou, and E. Menasalvas, "C-DBSCAN: Density-Based Clustering with Constraints," in *Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, 2007, pp. 216–223.

[10] K. Makino and T. Uno, "New Algorithms for Enumerating All Maximal Cliques," in *Algorithm Theory*, 2004, pp. 260–272.

[11] Yun Zhang, F. N. Abu-Khzam, N. E. Baldwin, E. J. Chesler, M. A. Langston, and N. F. Samatova, "Genome-Scale Computational Approaches to Memory-Intensive Applications in Systems Biology," in *Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, 2005.

[12] J. W. Moon and L. Moser, "On cliques in graphs," *Israel journal of Mathematics*, vol. 3, no. 1, pp. 23–28, 1965.

[13] X. Li, R. Zhou, L. Chen, Y. Zhang, C. Liu, Q. He, and Y. Yang, "Finding a Summary for All Maximal Cliques," in *Proceedings of the 37th IEEE International Conference on Data Engineering*, 2021, pp. 1344–1355.

[14] K. Murakami and T. Uno, "Efficient algorithms for dualizing large-scale hypergraphs," *Discrete Applied Mathematics*, vol. 170, pp. 83–94, 2014.

[15] V. V. Vazirani, *Approximation Algorithms*. Springer Science & Business Media, 2013.

[16] V. Chvatal, "A Greedy Heuristic for the Set-Covering Problem," *Mathematics of operations research*, vol. 4, no. 3, pp. 233–235, 1979.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[18] A. Mehrotra and M. A. Trick, "A column generation approach for graph coloring," *INFORMS Journal on Computing*, vol. 8, no. 4, pp. 344–354, 1996.

[19] D. Dua and C. Graff, "UCI Machine Learning Repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[20] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating Constraints and Metric Learning in Semi-Supervised Clustering," in *Proceedings of the 21st International Conference on Machine Learning*, 2004, pp. 839–846.