Clustering Under Radius Constraints Using Minimum Dominating Sets

Quentin Haenn, Brice Chardin, Mickaël Baron

Clustering Under Radius Constraints Using Minimum Dominating Sets

Quentin Haenn[1], Brice Chardin[1], and Mickael Baron[1]

ISAE-ENSMA, Université de Poitiers, LIAS, France
{quentin.haenn, brice.chardin, mickael.baron}@ensma.fr

Abstract. In this paper, we evaluate the applicability of algorithms designed to solve the minimum dominating set problem to perform clustering. The associated clustering problem relies on user constraints, and more specifically on radius intra-cluster constraints. We adapt and evaluate implementations from the state of the art on classification datasets, to compare them with other exact or approximate radius-based clustering algorithms, namely equiwide clustering and hierarchical agglomerative clustering with minimax linkage. We consequently provide the benchmark tools and datasets used in this work.

Keywords: Constrained Clustering · Radius Based Clustering · Minimum Dominating Set

1 Introduction

One of the major benefits of clustering is the ability to find groups and patterns that lie under the data, particularly without much knowledge of it [12]. Clustering is used in many fields, such as data mining, machine learning, pattern recognition, image analysis, bioinformatics, etc. [2, 8, 12, 14], and is a very active field of research.

Clustering under user constraints is a particular type of clustering, where users provide as input of the algorithm constraints that are coherent with their prior knowledge of the data [9]. Typical constraints are instance-based, i.e., users can either specify that two points must belong to the same cluster (must-link), or that two points must belong to different clusters (cannot-link). This has led to global, cluster-based constraints such as diameter and radius constraints. In this work, we focus on the clustering under radius constraints (CRC) problem, which is capable of finding clusters that satisfy a provided error bound and find an appropriate representative element for each cluster.

Since such constraints can lead to trivial solutions such as each points belonging to its own cluster, we add the global constraint that the number of clusters must be minimal. This is often desirable, especially when the user does not know how many clusters are originally present in the data.

Previous work has already introduced such a problem [1, 6], but has mainly focused on diameter constraints. Consequently, we found that the minimum dominating set (MDS) has been linked to the CRC problem [1–3, 11]. The MDS problem is a well-known NP-Hard graph problem, but recent advances in exact and

approximate algorithms have made it possible to solve it efficiently [4, 13]. As solving the MDS problem relies on heavy computations but theoretically offers optimal solutions to the CRC problem (i.e., the number of clusters is minimal), we propose to study the feasibility of this approach in comparison to state of the art clustering algorithms. If this approach is efficient, it could be a new way to solve the CRC problem, providing users a new tool to reduce the dimensionality of their data.

Industrial Use Case: We aim to apply this approach to the analysis of electrical consumers and producers along an electrical grid. This is an instance of dimension reduction, as we attempt to identify representative components on the grid. This step is part of a process to consistently reduce the computation time required for further modeling of the grid.

Paper organization In the remainder of this section we introduce necessary notions and definitions and formalize the clustering under radius constraint problem. Section 2 provides a brief overview of the CRC problem and the MDS problem. Section 3 presents a CRC algorithm based on MDS. We provide an experimental evaluation of state-of-the-art algorithms in section 4. We conclude by discussing the feasibility of this approach.

1.1 Preliminaries

A clustering task is concerned with finding a partition $\mathcal{P} = \{C_1, C_2, \ldots, C_n\}$ of a population $\mathcal{S}$ such that each cluster C_i is a subset of $\mathcal{S}$, and $\bigcup_{i=1}^{n} C_i = \mathcal{S}$. Assigning a point to a cluster is based on dissimilarity measures between points. We note $d(a, b)$ the dissimilarity between points a and b. Upon this definition, some authors introduced various wideness measures of a cluster, such as the diameter, the radius, the average distance, etc. [1–3, 6, 7, 9]. In this paper, we only consider the radius concept of wideness characterization of a subset of elements of a population.

The radius of a cluster C has been identified by Hubert [11] and applied by Ao et al. [2] as the minimum eccentricity within C: $R(C) = \min_{a \in C} \max_{b \in C} d(a, b)$. Based on this, the natural definition of the center (i.e. the representative point) of a cluster is the point a of C such that $a = \arg\min_{a \in C} \max_{b \in C} d(a, b)$. Intuitively, this center is the point with the best worst-case dissimilarity to any other point in the cluster: $\forall b \in C, d(a, b) \leq R(C)$.

Definition 1 (Clustering Under Radius Constraints Problem). *Let $\mathcal{S}$ be a population, d a dissimilarity function, and T a threshold. The clustering under radius constraints problem is the problem of finding a partition $\mathcal{P}$ of $\mathcal{S}$ such that:*

- $\forall C \in \mathcal{P}, R(C) \leq T$
- $\bigcup_{C \in \mathcal{P}} C = \mathcal{S}$
- *$|\mathcal{P}|$ is minimum, i.e., there is no partition $\mathcal{P}'$ such that $|\mathcal{P}'| < |\mathcal{P}|$ and $\mathcal{P}'$ is a solution to the initial CRC problem.*

Because we aim to apply graph based approaches to the CRC problem, we introduce related notions and definitions. A simple graph $G = (V, E)$ is a couple where V is a set of vertices and E is a set of edges. An edge $e = (u, v)$ is a pair of adjacent vertices u and v. A set of vertices $D \subseteq V$ is a dominating set of G if and only if $\forall v \in V \setminus D, \exists u \in D, (u, v) \in E$. The minimum dominating set problem is the problem of finding a dominating set $D \subseteq V$ of G such that $|D|$ is minimum.

2 Related Work

Radius-based approaches provide some benefits compared with other approaches, as they lead intuitively to a clustering that includes a representation of the data. This representative element is not available under diameter constraint, and sometimes not even computable [1]. This representation problem is an integral part of the clustering problem [12].

One approach to the CRC problem is the Equiwide Clustering (EQW) algorithm [1]. In this work, the authors propose an exact algorithm based on a linear programming (LP) formulation. Their experiments show that the algorithm is able to find the optimal solution in reasonable time on relatively small real world dataset (less than 2000 instances). However, this algorithm efficiency drops when applied to bigger datasets.

Another well-known approach is hierarchical agglomerative clustering (HAC) with the min max linkage criterion [2, 3], but it is also well-known that HAC is not designed to find the optimal solution regarding the minimality of the partition under a given constraint. To date, the most efficient HAC algorithm using MinMax criterion identified in the literature is Protoclust [3, 17].

The MDS problem has been proven linked to the CRC problem with a few adjustments [2, 11]. Thus, the possibility of using MDS to solve CRC problems has already been mentioned [2, 3] but not implemented due to the lack of efficient algorithms.

Lately, an approximate MDS algorithm has been proposed by Casado et al. [4]. The authors showed that the approximation is efficient on classical graph instances when compared to other state of the art approximate algorithms [5, 15, 16], and concluded that it may be interesting to use it on different combinatorial optimization problems.

Jiang and Zheng [13] released an exact algorithm to solve the MDS problem, based on the idea that the MDS problem can be solved by a novel branch and bound algorithm and bounded the search space thanks to the 2-hop adjacency of the graph, that is to say, two vertices are 2-hop adjacent if and only if they are adjacent or if there exists a vertex that is adjacent to both of them. Considering this definition, if two vertices are not 2-hop adjacent, then they cannot be in the same dominating set. According to Jiang and Zheng, it is the first efficient Branch-and-Bound exact algorithm to solve the MDS problem to date.

Following the previous state of the art, we propose to study implementations of CRC algorithms built on top of both the exact and approximate MDS

Table 1: Selected state of the art algorithms

Algorithm	Paradigm	Minimal #Clusters	Language
Protoclust [3]	MinMax HAC	No	R
EQW-LP [1]	LP	Yes	Python
MDS-APPROX [4]	MDS	No	Python + Java
MDS-EXACT [13]	MDS	Yes	Python + C

algorithms [4, 13], and to compare their efficiency with two state of the art algorithms: Protoclust [3] and Equiwide Clustering [1].

Table 1 lists the characteristics of the algorithms included in the experimental evaluation of this study.

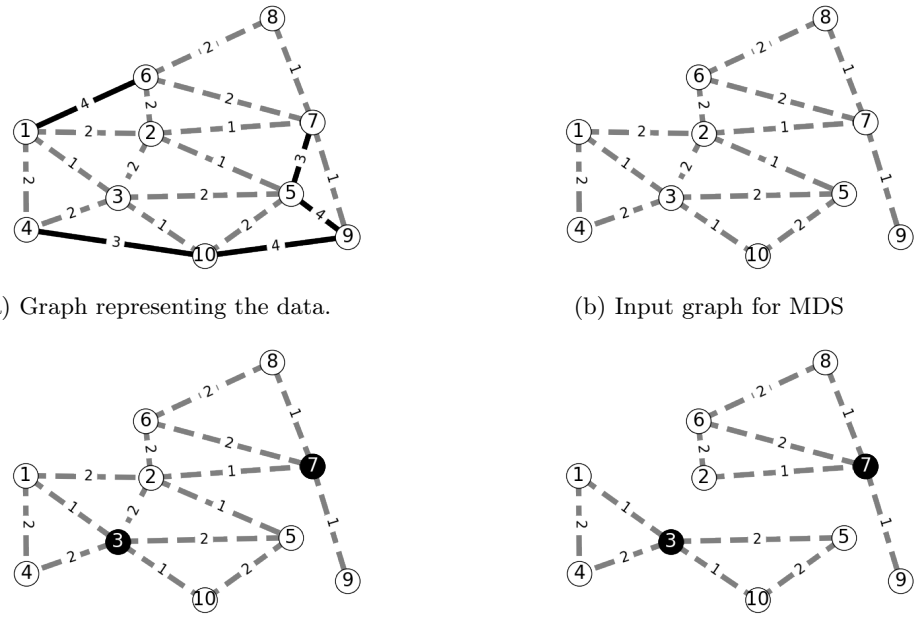
3 Minimum Dominating Set Based Clustering

A Ten-Point Example for MDS-Based Clustering Let $\mathcal{S}$ be a set of points that we want to cluster into $\mathcal{P}$. Let d be a dissimilarity function. Let us consider that the user already analyzed the data and found that the maximum admissible dissimilarity between a point and its representative is 2. Thus, the user wants to cluster the data into clusters under a radius constraint, or threshold, $T = 2$. We illustrate the MDS-based clustering algorithm on a simple example, shown in Fig. 1. Fig. 1a represents the input data transformed into an equivalent graph. Each vertex represents a point of the population, and each edge represents a dissimilarity between two points. The weight of the edge is the dissimilarity between the two points. For readability we did not represent each pairwise dissimilarity. Dark edges in Fig. 1a represent the edges whose weight exceeds the threshold ($T = 2$).

To convert the original clustering input, i.e: $\mathcal{S}$, d and T, into a suitable input for MDS, the initial equivalent graph is converted into a graph $G'(\mathcal{S}, E)$ where $E = \{(x_i, x_j) \mid d(x_i, x_j) \leq T\}$, i.e, over-weighted edges are removed. This graph is illustrated in Fig. 1b.

This graph is then provided as the input of MDS algorithms. Their output is a dominating set, i.e., a set of vertices. In the example represented in Fig. 1, the set $\{3, 7\}$ is the only MDS, because we cannot find a dominating set with less than two vertices. Using this MDS, we say that points 3 and 7 are representative elements, or centers, of their cluster.

To provide a clustering, this dominating set has to be transformed into a proper partition of the input population. Several solutions might be considered to assign a point to its cluster, as long as the affected point is dominated by the center of the cluster it is assigned to. The assignment selected for this evaluation is to group elements with the closest center, i.e., the center with which the dissimilarity is minimal. When a point is dominated by multiple representative points, it is assigned to the cluster of the closest representative point. In our example, 2 is dominated by both 3 and 7, as illustrated in Fig. 1c. Thus, it is



(a) Graph representing the data.

(b) Input graph for MDS

(c) Minimum Dominating Set

(d) Resultants Clusters With Their Representatives

Fig. 1: Illustration of the clustering process using the MDS approach

assigned to the cluster of 7, as $d(2,7) < d(2,3)$. An edge case can occur when a point is dominated by multiple representative points with the same dissimilarity. In this case, the point is assigned arbitrarily.

Once each vertex has been assigned to a cluster, the algorithm returns the clusters and their representative points. The clustering result is illustrated in Fig. 1d.

This example shows how the MDS problem solves the CRC problem.

4 Experiments

In this section we evaluate the algorithms previously listed in Table 1 on datasets from OpenML [18]. Their characteristics are described in Table 2.

The experiments are run on a Linux computer running Debian 12 with an Intel Core I5-10505 CPU at 3,20GHz, and 32GiB of DDR4 RAM. The Linux kernel is 6.1.0. The implementations of the algorithms based upon MDS approaches and EQW-LP are in Python. The approximative algorithm provided by [4] is implemented in Java. The exact algorithm, provided by [13] is implemented in C. Protoclust [3] is implemented in R. The python interpreter is version 3.11, the R interpreter version is 4.2.2 and the java platform is OpenJDK 17.0.9. The C

Table 2: Experimental Evaluation Datasets

Name	#Elements	#Dimensions	#Classes	R_{opt}
Iris	150	4	3	1.43
Wine	178	13	3	232.09
Glass Identification	214	9	6	3.94
Ionosphere	351	34	2	5.46
WDBC	569	30	2	1197.42
Synthetic Control	600	60	6	70.12
Vehicle	846	18	4	155.05
Yeast	1484	8	10	0.4235
Ozone	2534	72	2	245.59
Waveform	5000	40	3	10.74

compiler is gcc 12.2.0. The LP solver is Gurobi 10.0.0. Experiments are run ten times to evaluate the variance of the results. All the experiments are available online for reproducibility purposes [10].

4.1 Radius Threshold and Dissimilarity

Clustering under radius constraints requires a radius threshold and a dissimilarity function in addition to the datasets. We used the Euclidean Distance as the dissimilarity, in line with previous experimental evaluations on similar datasets [1, 6]. As for the radius threshold, it is not trivial to find an appropriate value without domain-specific knowledge. The same issue occurred in previous work for diameter constraints [6]. The authors proposed an iterative way of finding the optimal diameter from the number of classes taken as the desired number of clusters. Therefore, we adapted their work to find the optimal threshold to use in our experiments.

To do so, we performed a binary search on the threshold to use. The stopping criterion for that search is when the number of clusters returned by the CRC algorithm is the number of classes in the original dataset, and the immediate lower dissimilarity yields a greater number of clusters.

The optimal radius obtained from the binary search for each dataset are presented in the column R_{opt} of Table 2. Every radius is rounded up to 10^{-2} decimals for readability, except for the Yeast dataset due to its density. Nevertheless, the radius shown in Table 2 are the ones used in the experiments.

4.2 Results

Number of Clusters The first metric we analyzed is the number of clusters found by the algorithms. The results are presented in Table 3. First, the number of cluster is constant on all runs of all algorithms on all datasets. This supports the idea that the algorithms are stable. Second, exact algorithms (MDS-Exact and

Table 3: Number of cluster found by the algorithms on the datasets using the radius found

	MDS-Approx	MDS-Exact	EQW-LP	Protoclust
Iris	**3**	**3**	**3**	4
Wine	4	**3**	**3**	4
Glass Identification	7	**6**	**6**	7
Ionosphere	**2**	**2**	**2**	5
WDBC	**2**	**2**	**2**	3
Synthetic Control	8	**6**	**6**	8
Vehicle	5	**4**	**4**	6
Yeast	**10**	**10**	**10**	13
Ozone	3	**2**	**2**	3
Waveform	**3**	**3**	**3**	6

EQW-LP) always identify the optimal number of clusters while approximate algorithms might not. In addition, the number of clusters found by MDS-Approx is either exact or close to the optimal number of clusters, with at most two clusters difference noticed on the Synthetic Control dataset. Protoclust never reaches the optimal number of clusters, but remains relatively close to it with at most a three clusters difference for the Yeast dataset.

Effective Radius This metric allows to assess if the clusters found are valid, i.e., if the radius of the clusters is at most equal to the given constraint. Plus, this metric allows us to know if the clusters built are compact, i.e., if the effective maximal radius of the clusters is the minimal one that can be found under the constraint given. Results are displayed in Table 4 and optimal values are typeset in bold.

The effective radius is constant on all ten runs of all algorithms on all datasets. As the number of clusters before, this metric strengthens the idea that the algorithms are mostly stable.

The second thing we note is that, with MDS-Exact and EQW-LP, the effective radius is always equal to the constraint given. This confirms that, on one hand, the algorithms are optimal under this metric and, on the other hand, that the radius constraint is tight, namely that the MDS admits a single solution or, potentially, multiple solution but with the same resulting wideness. On the contrary, the effective radius given by Protoclust and MDS-Approx are sometimes smaller than the constraint given. This is directly linked to the number of clusters found by the implementation, because if the number of clusters is not optimal, the effective radius can be lower. This is always the case except on particular datasets such as Glass Identification, where the effective radius remains equal to the threshold despite the number of clusters being larger for MDS-Approx, and on Yeast dataset for Protoclust.

Lastly, we note that every algorithm satisfies the given radius constraint. This means that they are indeed all valid solutions to solve the CRC problem, although sometimes approximate w.r.t. the minimality criteria.

Table 4: Effective radius after clustering on the datasets

	MDS-APPROX	MDS-EXACT	EQW-LP	PROTOCLUST
Iris	**1.43**	**1.43**	**1.43**	1.24
Wine	220.05	**232.08**	**232.08**	181.35
Glass Identification	**3.94**	**3.94**	**3.94**	3.31
Ionosphere	**5.45**	**5.45**	**5.45**	5.35
WDBC	**1197.42**	**1197.42**	**1197.42**	907.10
Synthetic Control	66.59	**70.11**	**70.11**	68.27
Vehicle	150.87	**155.05**	**155.05**	120.97
Yeast	**0.423**	**0.423**	**0.423**	0.419
Ozone	235.77	**245.58**	**245.58**	194.89
Waveform	**10.73**	**10.73**	**10.73**	10.47

Execution Time Based upon the results presented in Table 5 we can split the analysis into two parts: exact algorithms and approximate algorithms.

Among exact algorithms, the MDS-Exact implementation is faster on small datasets, except on the ionosphere dataset, and is up to 4 times faster than EQW-LP, as observed on Synthetic Control. On the contrary, EQW-LP is faster on larger datasets, up to 100 times faster than MDS-Exact, as observed on Yeast. Based on these experiments, we conclude that EQW-LP is to be preferred on average, since its execution time remains comparable on small datasets, but becomes largely preferable with larger datasets under this metric.

As for approximate algorithms, MDS-Approx is faster than Protoclust on all datasets except Vehicle and Ozone, where it is 3 times slower. However, both algorithms run in mostly comparable execution times. Thus, considering MDS-Approx is either better or equivalent to Protoclust on the other metrics, we conclude that MDS-Approx is to be preferred on all datasets. By its design, we expected Protoclust to be faster on all datasets, however, this is mostly not the case. This is partially due to an implementation limitation of Protoclust because it always computes the entire dendrogram, to then cut it at the provided threshold. An improvement for this use case would be to stop the agglomerative clustering as soon as the threshold is reached.

Overall, the MDS-Approx algorithm becomes the preferred solution among all four with larger datasets when the minimality of the number of clusters is not to be guaranteed, since it becomes up to five times faster than exact solutions.

5 Conclusion and Future Work

In this work, we studied MDS-based clustering under radius constraint. Despite the fact that MDS approaches were identified very early on as being suitable for CRC problems, the lack of efficient algorithms, the inherent complexity of the problem itself and the hardware capabilities of computers meant that it was never applied. We have showed through various experiments that those approaches can indeed be considered as a tangible alternative to various CRC algorithms, both in terms of execution time and quality of the results. Plus, we showed that

Table 5: Execution time of the algorithms on the datasets in seconds

	MDS-APPROX	MDS-EXACT	EQW-LP	PROTOCLUST
Iris	0.062 ± 0.01	**0.009 ± 0.00**	0.018 ± 0.01	0.026 ± 0.00
Wine	0.029 ± 0.00	**0.010 ± 0.00**	0.014 ± 0.00	0.034 ± 0.00
Glass Identification	**0.015 ± 0.00**	0.020 ± 0.00	0.026 ± 0.00	0.046 ± 0.00
Ionosphere	**0.078 ± 0.01**	2.640 ± 0.05	0.104 ± 0.00	0.12 ± 0.00
WDBC	0.315 ± 0.01	**0.138 ± 0.00**	0.197 ± 0.01	0.402 ± 0.00
Synthetic Control	0.35 ± 0.03	**0.036 ± 0.00**	0.143 ± 0.01	0.489 ± 0.00
Vehicle	0.955 ± 0.04	**0.185 ± 0.00**	0.526 ± 0.01	0.830 ± 0.01
Yeast	**2.361 ± 0.03**	622.87 ± 0.30	6.718 ± 0.02	2.374 ± 0.08
Ozone	49.82 ± 1.18	1350.86 ± 1.5	26.86 ± 0.63	**15.32 ± 0.15**
Waveform	**48.01 ± 0.39**	5559.9 ± 15.3	233.9 ± 1.45	61.27 ± 0.08

both the exact and approximate variants of the algorithm are very efficient on small datasets, and that the approximate variant remains competitive on larger datasets. This work also may be considered as a first usage of Casado et al.[4] and Jiang and Zheng[13] algorithms on real combinatorial problem.

We conclude that the MDS approach seems promising and recent advances in this field mean this paradigm can be seen as a real alternative to classical algorithms or those requiring heavy and proprietary solvers such as linear programming.

Acknowledgments. This work was supported by the french region Nouvelle Aquitaine and the aLIENOR ANR LabCom (ANR-19-LCV2-0006). We also thank Jiang and Zheng for kindly providing us with the source code of their exact algorithm.

Disclosure of Interest The authors declare that they have no conflict of interest.

Bibliography

[1] Andersen, J., Chardin, B., Tribak, M.: Clustering to the Fewest Clusters Under Intra-Cluster Dissimilarity Constraints. In: 2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI), pp. 209–216 (Nov 2021), https://doi.org/10.1109/ICTAI52525.2021.00036

[2] Ao, S.I., et al.: CLUSTAG: hierarchical clustering and graph methods for selecting tag SNPs. Bioinformatics **21**(8), 1735–1736 (Apr 2005), https://doi.org/10.1093/bioinformatics/bti201

[3] Bien, J., Tibshirani, R.: Hierarchical Clustering With Prototypes via Minimax Linkage. Journal of the American Statistical Association **106**(495), 1075–1084 (Sep 2011), ISSN 0162-1459, https://doi.org/10.1198/jasa.2011.tm10183

[4] Casado, A., et al.: An iterated greedy algorithm for finding the minimum dominating set in graphs. Mathematics and Computers in Simulation

207, 41–58 (May 2023), ISSN 0378-4754, https://doi.org/10.1016/j.matcom.2022.12.018
[5] Chalupa, D.: An order-based algorithm for minimum dominating set with application in graph mining. Information Sciences **426**, 101–116 (Feb 2018), ISSN 0020-0255, https://doi.org/10.1016/j.ins.2017.10.033
[6] Dao, T.B.H., Duong, K.C., Vrain, C.: Constrained clustering by constraint programming. Artificial Intelligence **244**, 70–94 (Mar 2017), ISSN 00043702, https://doi.org/10.1016/j.artint.2015.05.006
[7] Dinler, D., Tural, M.K.: A Survey of Constrained Clustering. In: Unsupervised Learning Algorithms, pp. 207–235, Cham (2016), ISBN 978-3-319-24211-8, https://doi.org/10.1007/978-3-319-24211-8_9
[8] Gao, Z., et al.: Multi-level aircraft feature representation and selection for aviation environmental impact analysis. Transportation Research Part C: Emerging Technologies **143**, 103824 (Oct 2022), ISSN 0968-090X, https://doi.org/10.1016/j.trc.2022.103824
[9] Gordon, A.D.: A survey of constrained classification. Computational Statistics & Data Analysis **21**(1), 17–29 (Jan 1996), ISSN 0167-9473, https://doi.org/10.1016/0167-9473(95)00005-4
[10] Haenn, Q., Chardin, B., Baron, M.: MDS clustering Experiments. Source Code repository (2024), https://forge.lias-lab.fr/mds_clustering
[11] Hubert, L.J.: Some applications of graph theory to clustering. Psychometrika **39**(3), 283–309 (Sep 1974), ISSN 1860-0980, https://doi.org/10.1007/BF02291704
[12] Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Computing Surveys **31**(3), 264–323 (Sep 1999), ISSN 0360-0300, 1557-7341, https://doi.org/10.1145/331499.331504
[13] Jiang, H., Zheng, Z.: An Exact Algorithm for the Minimum Dominating Set Problem. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, pp. 5604–5612 (Aug 2023), ISBN 978-1-956792-03-4, https://doi.org/10.24963/ijcai.2023/622
[14] Liu, Y., Sioshansi, R., Conejo, A.J.: Hierarchical Clustering to Find Representative Operating Periods for Capacity-Expansion Modeling. IEEE Transactions on Power Systems **33**(3), 3029–3039 (May 2018), ISSN 1558-0679, https://doi.org/10.1109/TPWRS.2017.2746379
[15] Potluri, A., Singh, A.: Two Hybrid Meta-heuristic Approaches for Minimum Dominating Set Problem. In: Swarm, Evolutionary, and Memetic Computing, pp. 97–104 (2011), ISBN 978-3-642-27242-4, https://doi.org/10.1007/978-3-642-27242-4_12
[16] Potluri, A., Singh, A.: Hybrid metaheuristic algorithms for minimum weight dominating set. Applied Soft Computing **13**(1), 76–88 (Jan 2013), ISSN 1568-4946, https://doi.org/10.1016/j.asoc.2012.07.009
[17] Tai, X.H., Frisoli, K.: Benchmarking Minimax Linkage (Jun 2019), arXiv:1906.03336 [cs, stat]
[18] Vanschoren, J., van Rijn, J.N., Bischl, B., Torgo, L.: Openml: networked science in machine learning. SIGKDD Explorations **15**(2), 49–60 (2013), https://doi.org/10.1145/2641190.2641198