Partitioning a Map into Homogeneous Contiguous Regions: A Branch-And-Bound Approach Using Decision Diagrams

Nicolas Golenvaux
Institute for Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), Université catholique de Louvain, Louvain-la-Neuve, Belgium

Xavier Gillard
Institute for Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), Université catholique de Louvain, Louvain-la-Neuve, Belgium

Siegfried Nijssen
Institute for Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), Université catholique de Louvain, Louvain-la-Neuve, Belgium

Pierre Schaus
Institute for Information and Communication Technologies, Electronics and Applied Mathematics (ICTEAM), Université catholique de Louvain, Louvain-la-Neuve, Belgium

Abstract

Regionalization is a crucial spatial analysis technique used for partitioning a map divided into zones into k continuous areas, optimizing the similarity of zone attributes within each area. This technique has a variety of applications in fields like urban planning, environmental management, and geographic information systems. The REDCAP algorithm is a well-known approach for addressing the regionalization problem. It consists of two main steps: first, it generates a spatially contiguous tree (SCT) representing the neighborhood structure of the set of spatial objects using a contiguity-constrained hierarchical clustering method. Second, it greedily removes $k-1$ edges from the SCT to create k regions. While this approach has proven to be effective, it may not always produce the most optimal solutions. We propose an alternative method for the second step, an exact dynamic programming (DP) formulation for the k-1 edges removal problem. This DP is solved using a multi-valued decision diagram (MDD)-based branch and bound solver leading to a more optimal solution. We compared our proposed method with the REDCAP state-of-the-art technique on real data and synthetic ones, using different instances of the regionalization problem and different supervised and unsupervised metrics. Our results indicate that our approach provides higher quality partitions than those produced by REDCAP at acceptable computational costs. This suggests that our method could be a viable alternative for addressing the regionalization problem in various applications.

2012 ACM Subject Classification Computing methodologies → Discrete space search

Keywords and phrases Regionalization, Redcap, Skater, Multivalued Decision Diagrams

Digital Object Identifier 10.4230/LIPIcs.CP.2023.45

Category Short Paper

1 Introduction

Spatial analysis plays a crucial role in comprehending and managing intricate spatial relationships [13]. A fundamental challenge in spatial analysis involves determining homogeneous regions based on similarity computed from shared attributes.

Given a geographical map divided into zones that partition the space, each zone is associated with a set of attributes (e.g., population density, land use, socio-economic factors, etc.) as represented on Figure 1a where the colors represent the attributes. The regionalization

29th International Conference on Principles and Practice of Constraint Programming (CP 2023).
Editor: Roland H. C. Yap; Article No. 45; pp. 45:1–45:10
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

problem studied in this paper, involves grouping these zones into k contiguous areas (also referred to as regions), optimizing the similarity of attributes within each area. The contiguity constraint requires that each grouping forms a single, connected area without any isolated parts or exclaves.

Solving the regionalization problem can be computationally complex, even for a moderate number of zones and areas, as it often entails searching for an optimal solution within a large combinatorial search space. The state-of-the-art method called REDCAP [12] involves solving the problem in a two-step approach as illustrated on Figure 1. In the first step, a spatially contiguous (spanning) tree is created using a hierarchical clustering strategy. In the second step, $k-1$ edges are greedily deleted from the tree. The final contiguous areas are defined by the k remaining connected components of the tree, ensuring both attribute similarity and contiguity.

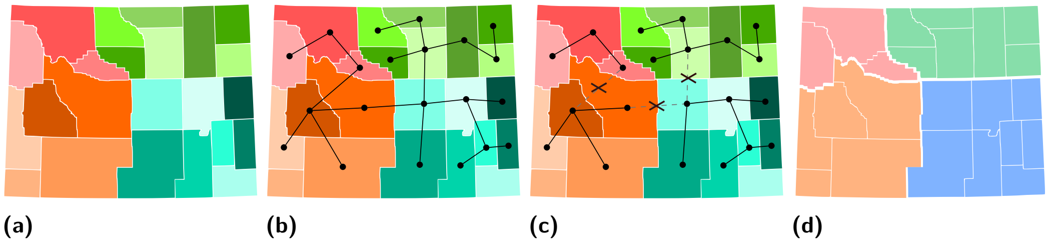

Figure 1 Steps for solving a regionalization problem to cluster four areas: (a) Input of the problem where colors represent the attributes of each zone. (b) Create a spatially contiguous tree that connects all the zones. (c) Remove three edges from the tree. (d) The output areas are then formed.

2 Related Work

The regionalization problem has been a well-studied combinatorial problem since the 1970s [5]. Regionalization methods can be broadly classified as spatially implicit or spatially explicit models, depending on how they represent the spatial contiguity constraints of the formed regions [9]. Implicit methods initially apply traditional or non-spatial clustering methods to obtain a preliminary solution, which is then adjusted to enforce spatial constraints [16, 17]. Conversely, explicit models enforce spatial contiguity constraints from the outset [9].

Exact methods provide an optimal guarantee for the solutions. However, they are considered computationally intensive and still limited to small problems [8], meaning they are suited for situations with a low number of zones and regions. In contrast, heuristic approaches, which are more scalable, do not guarantee optimal solutions.

Among heuristic approaches with spatially explicit constraints, tree-based methods such as SKATER [14, 2] and REDCAP [12] are widely used and have been demonstrated to generate near-optimal partitions with acceptable computational costs [1, 6]. Both SKATER and REDCAP employ a two-step approach that first constructs a spatially contiguous tree connecting all the zones, then greedily splits it to create the desired number of regions.

Certain spatially explicit regionalization methods, such as those described by [7] and [18], do not require a predetermined number of regions. Instead, these methods aim to identify underlying regions while imposing constraints related to these regions.

It is worth noting that the regionalization problem can be viewed as a variant of the optimal graph partitioning problem [4]. The main difference is that, in graph partitioning, the dissimilarities between non-adjacent pairs of nodes are not considered by the objective function.

3 Proposed Approach

We start by stating formally the problem. Let us denote by $V = \{1, 2, \dots, n\}$ the zones of our map or more generally the zones. Let $G = (V, E)$ represent the contiguity graph of the map, where edges $(i, j) \in E$ exist if and only if zones i and j share a common border on the map. G must be connected. Let $P = \{V_1, V_2, \dots, V_k\}$ represent a partition (areas) of the zones V into k regions, with $1 \leq k \leq n$. A partition P is considered feasible if all areas are disjoint, cover the original set of zones, and the induced subgraphs $G(V_u)$ are connected for all $u \in \{1, 2, \dots, k\}$. Let $(a_{i,1}, a_{i,2}, \dots, a_{i,m})$ denote the m numerical attributes of zone i. The quadratic distance or dissimilarity between zones based on the set of numerical attributes is $d_{i,j} = \sum_{l=1}^{m}(a_{i,l} - a_{j,l})^2$. The *heterogeneity* of a region V_u is defined as $h(V_u) = \frac{1}{|V_u|}\sum_{i<j|i,j\in V_u} d_{i,j}$. It can be shown that the heterogeneity can equivalently be computed as the sum of squared distances to the mean of attributes $h(V_u) = \sum_{i\in V_u}\sum_{l=1}^{m}(a_{i,l} - \bar{a}_l)^2$ with $\bar{a}_l = \frac{1}{|V_u|}\sum_{i\in V_u} a_{i,l}$. The regionalization problem is to find a feasible k-partition minimizing the overall heterogeneity $H(P) = \sum_{u=1}^{k} h(V_u)$.

3.1 The REDCAP Two-Step Approach

REDCAP [12] is a state-of-the-art method for solving the regionalization problem. This approach consists of two consecutive steps. First, it identifies a spanning tree, T, of graph G (also referred to as a spatially contiguous tree (SCT) in this context) using a hierarchical clustering approach. Second, it identifies $k - 1$ edges that partition the tree into a forest of k subtrees, each of which constitutes the final cluster of the regionalization problem.

Step1. Starting initially with a set of n clusters C, each one containing one of the zones $C = \{c_1 = \{1\}, c_2 = \{2\}, \dots, c_n = \{n\}\}$, the hierarchical clustering approach merges at each step, the two closest contiguous clusters c_I and c_J until one single cluster regroups all the zones. Two clusters c_I and c_J are considered contiguous if there is an edge in the connectivity graph G linking two zones from each cluster c_I and c_J. The distance between two clusters c_I and c_J denoted as $D(c_I, c_J)$ and can be computed with different variants. The variant that generally yields the best results is called full-order complete linkage (Full-Order-CLK) defined as $D(c_I, c_J) = \max\limits_{i\in c_I, j\in c_J} d_{i,j}$. Initially empty, one edge is thus added to the SCT T each time two clusters c_I and c_J are merged. This edge $e \in E$ is the one of the original connectivity graph with minimal cost i.e. $\text{argmin}_{(i,j)\in E|i\in c_I, j\in c_J} d_{i,j}$. At the end of the procedure, T contains $n - 1$ edges connecting all the nodes. Overall, the computational complexity for building T using the aforementioned method is $O(n^2 \log n)$.

Step2. The second step of REDCAP [12] to obtain k homogeneous regions is to identify $k - 1$ edges to remove from T as illustrated in Figure 1c. The remaining components form the final k regions. Due to the inherent complexity of finding an optimal solution for the second-step tree partitioning problem [14], this problem is solved in REDCAP using a greedy heuristic. Let us denote by $F = \{T_1, T_2, \dots, T_k\}$ the spanning forest obtained after the removal of $k - 1$ edges from T. The set of nodes and edges of each tree T_u are denoted by V_u and E_u.

At each iteration, one edge is taken out, splitting one tree of the forest into two trees. Notice that a subtree can possibly contain a single node in case a leaf-edge is removed. The edge that results in the greatest decrease in heterogeneity (or in other words, the highest *homogeneity gain*) is chosen to be eliminated.

For a tree $T_u = (V_u, E_u)$, the homogeneity gain $h_g(e)$ obtained by the removal of an edge $e \in E_u$, dividing the tree into two trees T_{u_1} and T_{u_2} is defined as $h_g(e) = h(V_u) - h(V_{u_1}) - h(V_{u_2})$ where V_{u_1}, V_{u_2} are the nodes in the corresponding sub-trees T_{u_1} and T_{u_2}. The complexity of this greedy algorithm is $O(k \cdot n^2)$ but as k is usually much smaller than n, it can be ignored. We propose to replace this second-step greedy algorithm for the tree partitioning problem by an exact formulation using dynamic programming and MDD-based optimization as explained in the next section.

3.2 Edge Removal using MDD-based Optimization

We express the problem of the optimal removal of $k-1$ edges from the spanning tree $T = (V_T, E_T)$ to minimize the heterogeneity as a Dynamic Programming Problem. We use a sequence of $k-1$ decision variables x_i representing the edges that are successively removed from the SCT. The domain of each of these variables is the set of edges E_T of the tree. The search-space can be described as Layered Transition Diagram, also called Multivalued Decision Diagrams (MDD) [3]. Let us denote by F_i the set of possible forests obtained by removing exactly i edges from T. This set of forests $\bigcup_{0 \leq i \leq k-1} F_i$ constitutes the state space and the corresponding nodes of the MDD. A state (forest) is denoted by $f = (V_f, E_f)$. Since we always remove edges on the transitions, but not nodes, the set of nodes of each forest of the state space remains the one of the original contiguity graph $V_f = V, \forall f \in F_i, \forall 0 \leq i \leq k-1$. Let us now describe the important nodes, the transition function and cost functions for the MDD:

- The set of state-spaces $F = \{F_0, \ldots, F_{k-1}\}$ forms the layers, where F_i corresponds to all the states formed by removing exactly i edges from T.
- The root of the MDD is denoted as r and corresponds to the state $f^0 \in F_0$, with $f^0 = T$, and its initial value is $v_r = -h(V_T)$, representing the heterogeneity of the entire original map.
- The terminal states are denoted by t and regroup every state $f^{k-1} \in F_{k-1}$.
- The set τ of transition functions s.t. $\tau_i : F_i \times E \to F_{i+1}$ for $i = 0, \ldots, k-2$ taking the system from one state f^i to the next state f^{i+1} based on the edge removed.
- The set c of transition cost functions $c_i : F_i \times E \to \mathbb{R}$ s.t. $c_i(f, e)$ is the homogeneity gain $h_g(e)$ of making the decision for x_i to remove the edge e from the forest f at the level i.

The objective function is then to maximize $v_r + \sum_{i=0}^{k-2} c_i(f^i, x_i)$ so that $f^{i+1} = \tau_i(f^i, x_i)$ and $x_i \in E_T, \forall i \in \{0, \ldots, k-2\}; f^i \in F_i, \forall i \in \{0, \ldots, k-1\}$. The optimal solution can be obtained by searching the longest path from the root r to one of the terminal nodes t.

3.3 Branch-and-Bound with MDD

The number of states in the MDD augments rapidly with k and n ($n-1$ choose $k-1$). For such a situation, Bergman et al. [3] have introduced a branch-and-bound (BnB) framework to explore the state space of the MDD without generating it completely upfront and keeping the memory requirement limited. In BnB based on MDDs, relaxed and restricted MDDs, obtained by limiting the width of the MDDs, are used to efficiently explore and prune the solution space. A relaxed MDD is obtained by state-merging. It is an over-approximation of the solution space, where some infeasible solutions might be included. The optimal path of a relaxed MDD provides an upper-bound, which can be used to prune the search space. A restricted MDD, on the other hand, is an under-approximation of the solution space. It is obtained by discarding the less promising states. Some feasible solutions might be excluded but it can nevertheless be used to get lower bounds (similarly to a beam-search). MDD based

BnB enqueues nodes of the original MDD in the queue. When a node is popped, a restricted and a relaxed MDD are compiled from this node to hopefully improve the incumbent solution and prune the search by upper-bounding. This combination of dynamic compilation of restricted and relaxed MDD allows for a more effective exploration of the solution space of the MDD and helps to find the optimal solution in a computationally efficient manner. The nodes can also be pruned by computing a (cheap) upper-bound [11]. We describe next the state-merging procedure and the cheap upper-bound for the optimal edge removal problem.

3.3.1 State Merging

As opposed to restricted MDD, a relaxed MDD encodes a superset of the solutions of the original MDD and thus leads to an upper-bound. Its construction is limited to a given width by applying a problem-specific merge operator. In the context of the edge-removal problem, the merge operator applied to two nodes at a same level simply consists in taking the union of the edges in the two forests: $\texttt{merge}(f_A = (V, E_{f_A}), f_B = (V, E_{f_A})) = (V, E_{f_A} \cup E_{f_B})$. The costs of the arcs leading to the merged nodes remain unchanged. Notice that in a relaxed MDD, it is no longer true that the forests f^i at level i have exactly $i+1$ components. One can be convinced that this merging definition correctly includes a superset of the possible paths. It also guarantees an upper-bound on the optimal homogeneity gain that would be obtained without compression. This is a direct consequence of the following property.

▶ **Lemma 1.** *For a tree $T = (V, E)$ and a super-tree of T denoted $T' = (V', E')$ with $V \subseteq V'$ and $E \subseteq E'$, let e be an edge present in both E and E'. The homogeneity gain of this edge removal in T denoted $h_g(e)$ is lower than $h'_g(e)$ i.e. when this edge is removed from T'.*

Proof. Assuming e connects the two sub-trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ of T and $T'_1 = (V'_1, E'_1)$ and $T'_2 = (V'_2, E'_2)$ of T'. The homogeneity gain in T is $h_g(e) = h(V) - h(V_1) - h(V_2)$ which can equivalently be computed as $h_g(e) = \sum_{i \in V_1} \sum_{j \in V_2} d_{i,j}$. Similarly $h'_g(e) = \sum_{i \in V'_1} \sum_{j \in V'_2} d_{i,j}$. Therefore $h'_g(e) - h_g(e) = \sum_{i \in V'_1 \setminus V_1} \sum_{j \in V'_2 \setminus V'_1} d_{i,j} \geq 0$. ◀

3.3.2 Cheap Upper-Bound

To efficiently compute an upper-bound for a state forest f^i at level i, one can assume that the $k - i - 1$ remaining edges that will be removed induce sub-trees that are perfectly homogeneous (null heterogeneity). The upper-bound on the total homogeneity gain starting from f^i can then be calculated as : $\sum_{l=1}^{\min(k-i-1, |f^i|)} h(V_l^*)$ where $V_1^*, \ldots, V_{k-i-1}^*$ are the $k - i - 1$ trees of f^i having the highest heterogeneity value. Notice that the count of trees in f^i may be fewer than $k - i - 1$. Should this occur, the formula accounts for the heterogeneity of all present trees in the state.

4 Experiments

To assess the effectiveness of our MDD-based method for partitioning spatially contiguous trees, we carried out experiments that compare our approach to REDCAP using both real-life datasets and synthetic datasets with known ground-truth regions.

Real-life datasets. We use a set of 5 real-life regionalization datasets varying in size, geometry and number of attributes. Since comparing the absolute value of heterogeneity is meaningless, we use the *Rescaled Overall Heterogeneity H_r*. That corresponds to the ratio between the overall heterogeneity of the MDD approach and the one of REDCAP.

- **Economic and demographic indicators in NUTS zones:** We collected economic and demographic data on the NUTS areas of Europe to create several datasets that can be used as instances of the regionalization problem. Using the Eurostat database, we gathered data on the density, median age, average GDP per inhabitant, and migration rate for each NUTS-1, NUTS-2, and NUTS-3 European zones in 2019. We constructed one regionalization dataset for each level of NUTS classification, which we refer to as *Ecodemo NUTS1*, *Ecodemo NUTS2*, and *Ecodemo NUTS3*. After removing the unconnected NUTS zones, we are left with 94 zones for the *Ecodemo NUTS1* dataset, 236 for the *Ecodemo NUTS2* dataset, and 1,155 for the *Ecodemo NUTS3* dataset, each having four attributes.
- **Education in Belgium:** We collected data on the level of education in each municipality in Belgium for the year 2017 from the StatBel Open Data. Using this data, we created a regionalization dataset named *Education BE*, where the Belgian municipalities serve as the zones, and their attributes include the share of low, medium, and highly educated inhabitants living in their respective territories. This dataset comprises a total of 563 zones, each having three attributes.
- **USA Ecoregions:** Ecoregions are geographic regions of ecological systems based on vegetation, climate conditions, and land cover [15]. They are frequently used in conservation ecology for planning urban and agricultural development while preserving biodiversity. We use the same dataset as [1] to evaluate our regionalization model. The dataset gathers climatic and land-cover measurements from 1994 in 186 zones of the USA territory. Once the isolated zones are removed, our *Ecoregions USA* dataset includes 172 zones, each one having 15 different ecological attributes.

Synthetic Maps. We evaluate the performance of regionalization models in recovering the original regions on synthetic maps using supervised-learning metrics. Our synthetic maps are generated following the methodology proposed in [1], and are parameterized by the number of zones (cells of a square grid), number of regions, region fuzziness, and region geometries.

We created 3 different classes of synthetic maps obtained for different settings of the parameters of the generation process. Each class describes a different level of complexity for regionalization methods. The family A regroups maps with 100 zones divided in 5 regions with simple concentric geometries and well-delimited attribute values. The family B consists of synthetic maps of 400 zones distributed in 10 regions, with more complex and different geometries, and less pronounced frontiers between regions due to their more similar mean values. Finally, the family C comprises maps with 900 zones divided in 20 regions having more complex geometries and more diverse sizes but similar fuzziness than family B. An example of map for each family is represented on Figure 2.

In the case of synthetic datasets, we have access to the original partition, or the ground-truth. Consequently, the optimization problem can be reframed as a machine learning problem, with the goal of recovering the original partition. By comparing the assigned regions with the ground truth, we can calculate various machine learning metrics to assess the performance of the regionalization models. We use the pairwise comparison to evaluate the regionalization algorithms. Each pair of zones is labeled as positive if they belong in the same original region and as negative if they come from different ones. One can then compute the classical True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN) rates and evaluate the *precision*, *recall* and F_1 *score* metrics to evaluate the regionalization methods on the synthetic maps. In addition to these 3 supervised metrics, we also calculate the ratio between the overall heterogeneity of the partition generated by the regionalization algorithm with the one of ground-truth partition. We name this metric the *True Overall Heterogenity Ratio* H_T.

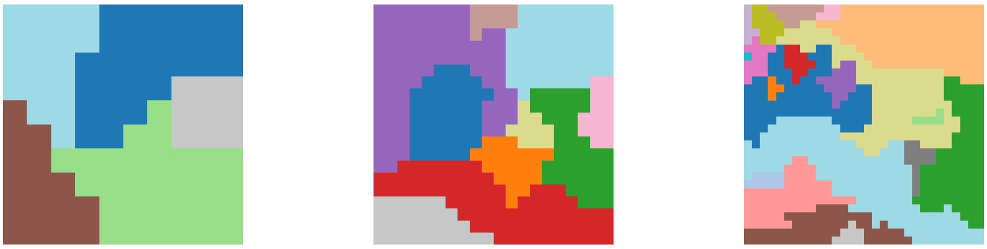

Figure 2 Examples of ground-truth regions for each synthetic map class. From left to right, we have an example of the regions' geometry of a synthetic map from family A, family B and family C. The zones belonging to the same region are colored in the same color.

Table 1 Comparison of MDD and REDCAP approaches on real-life regionalization problems. The table presents the rescaled overall heterogeneity for each dataset and partition size ($k \in 5, 10, 15, 20$).

	Number of regions k			
Dataset	5	10	15	20
Ecodemo NUTS1	1.00	0.98	0.96	0.965
Ecodemo NUTS2	0.989	0.97	0.958	0.935
Ecodemo NUTS3	0.997	1.00	1.00	1.00
Education BE	0,956	0.861	0.887	0.916
Ecoregions USA	1.00	0.997	0.963	0.953

5 Results

For each regionalization problem, we construct the corresponding SCT using REDCAP's hierarchical clustering. In the second step, we compare the greedy edge removal of REDCAP with the one computed using MDD-based optimization. We employ the DDO solver [10] with a width of 50 for both restricted and relaxed DDs and set a timeout of 100 seconds to identify the final k regions. Before the regionalization process, we normalize the attributes of the zones within a range between 0 and 1 using a MinMax scaler.

5.1 Real-life Instances

We assessed the performance of the MDD approach and REDCAP in generating 5, 10, 15, and 20 regions for each dataset described. The comparison between the two methods on the real-life regionalization problems is presented in Table 1. The table provides insights into the quality of the methods' partitions by displaying the rescaled overall heterogeneity for each dataset and for the partition sizes $k = 5, 10, 15, 20$.

Regardless of the requested number of regions, our experimental results show that the MDD approach consistently matches or outperforms REDCAP in terms of overall heterogeneity for each regionalization dataset. This indicates that the partitions generated by the MDD approach are of higher quality. However, the degree of difference varies depending on the dataset. For instance, in the case of the Education BE dataset, the MDD approach reduces the heterogeneity of the partition by almost 15% for 10 regions compared to REDCAP. In contrast, for the Ecodemo NUTS3 dataset, the two algorithms produce similar partitions for all values of k. Although the MDD approach only achieved optimality for the Ecodemo NUTS1 dataset when $k = 5$, it outperformed REDCAP for all datasets.

Table 2 Number of seconds taken by the MDD approach to find the first solution for each real-life regionalization problem. The time at which it discovers the best solution is presented in parentheses in the cases where it is not equivalent to the first solution founded.

	Number of regions k			
Dataset	5	10	15	20
Ecodemo NUTS1	<1	<1	<1	<1
Ecodemo NUTS2	<1	<1	2	2 (28)
Ecodemo NUTS3	6	13	19	21
Education BE	2 (18)	3 (85)	5 (97)	6
Ecoregions USA	<1	<1	<1 (21)	2 (24)

Table 3 Comparison of our MDD approach with REDCAP on synthetic datasets

	A		B		C	
Metric	MDD	REDCAP	MDD	REDCAP	MDD	REDCAP
Precision	**0,988**	0,944	**0,966**	0,912	**0,956**	0,954
Recall	**0,989**	0,962	**0,949**	0,891	**0,951**	0,908
F_1 Score	**0,988**	0,952	**0,956**	0,903	**0,954**	0,928
H_T	**0,99**	1,19	**0,978**	1,108	**1,015**	1,078

Regarding performance, there is a significant difference between the MDD approach and REDCAP. REDCAP takes, on average, between 0.01 and 0.3 seconds to produce a partition depending on the dataset and the number of regions requested, while the MDD approach is allowed to use up to 100 seconds to obtain the best possible solution. However, for the most part, the first partition discovered by the MDD approach is also the best one obtained within the 100-seconds timeframe. Table 2 presents the amount of time the MDD approach searched before finding the first valid partition for each regionalization problem. It is noteworthy that this first partition found by the MDD approach has always a lower or equal overall heterogeneity than the partition found by REDCAP. Additionally, if this first solution is not the best one found within the 100-seconds timeframe, Table 2 reports in parentheses the time taken by the MDD approach to discover the partition with the lowest overall heterogeneity.

5.2 Synthetic Maps

We evaluated both the REDCAP and MDD approaches on various synthetic maps of different sizes and complexities (families A, B, and C). For each family, we generated and assessed 20 maps using both methods. Table 3 displays the mean Precision, Recall, F1 Score, and True Overall Heterogeneity Ratio for each method, computed using the ground-truth values. The MDD approach was able to prove the optimal solution for the edge removal problem only for instances belonging to family A.

We can see that using the MDD-based approach for the second step of REDCAP improves the solution in all metrics for the three synthetic map families. Comparing the results of the MDD approach, it can be seen that the precision and recall are lower for families B and C. Both of these families share a characteristic in that their region boundaries are more fuzzy than those of family A. This suggests that our method encounters more difficulty in recovering the initial partition when the attributes between two neighboring regions are more similar, i.e. when the delimitations between regions are less pronounced. The number of zones, the number of regions and their geometric complexity seem to have a lesser impact on the capacity of our model to recover the original regions.

Moreover, for families A and B, the MDD approach generates regions with a lower overall heterogeneity than the ground-truth partition on average. This implies that the original partition is not always the optimal one in terms of overall heterogeneity. Thus, it can be concluded that for the first two families, the regions generated by the MDD approach deviate from the ground-truth ones simply because it discovered partitions with lower overall heterogeneity than the original ones.

6 Conclusion

In this paper, we have proposed a novel approach for the regionalization problem, an essential clustering task in a variety of spatial analysis domains. We have improved upon the second step of the well-established REDCAP algorithm by introducing an exact dynamic programming formulation for the edge removal problem solved using a multi-valued decision diagram (MDD)-based branch and bound solver. We have provided comprehensive experiments on both real-life datasets and synthetic ones to illustrate the efficacy of our method. Our comparison with the REDCAP algorithm using a wide range of supervised and unsupervised metrics demonstrated that our approach consistently produces partitions of higher quality.

References

1 Orhun Aydin, Mark Janikas, Renato Assunção, and Ting-Hwan Lee. A quantitative comparison of regionalization methods. *Int. J. Geogr. Inf. Sci.*, 35:1–29, 2021. doi:10.1080/13658816.2021.1905819.

2 Orhun Aydin, Mark V Janikas, Renato Assunçao, and Ting-Hwan Lee. Skater-con: Unsupervised regionalization via stochastic tree partitioning within a consensus framework using random spanning trees. In *Proceedings of the 2nd ACM SIGSPATIAL international workshop on AI for geographic knowledge discovery*, pages 33–42, 2018.

3 David Bergman, Andre A. Cire, Willem-Jan van Hoeve, and J. N. Hooker. Discrete optimization with decision diagrams. *INFORMS Journal on Computing*, 28(1):47–66, 2016. doi:10.1287/ijoc.2015.0648.

4 Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. *Recent advances in graph partitioning.* Springer, 2016.

5 Andrew D Cliff and Peter Haggett. On the efficiency of alternative aggregations in region-building problems. *Environment and Planning A*, 2(3):285–294, 1970.

6 Diep Dao and Jean-Claude Thill. Detecting Attribute-Based Homogeneous Patches Using Spatial Clustering: A Comparison Test. In *Information Fusion and Geographical Information Systems*, pages 37–54. January 2018. doi:10.1007/978-3-319-59539-9_4.

7 Juan C. Duque, Luc Anselin, and Sergio J. Rey. THE MAX-P-REGIONS PROBLEM*. *Journal of Regional Science*, 52(3):397–419, August 2012. doi:10.1111/j.1467-9787.2011.00743.x.

8 Juan C Duque and Richard L Church. A new heuristic model for designing analytical regions. In *North American Meeting of the International Regional Science Association, Seattle*, 2004.

9 Juan Carlos Duque, Raúl Ramos, and Jordi Suriñach. Supervised Regionalization Methods: A Survey. *International Regional Science Review*, 30(3):195–220, July 2007. Publisher: SAGE Publications Inc. doi:10.1177/0160017607301605.

10 X. Gillard, P. Schaus, and V. Coppé. Ddo, a generic and efficient framework for mdd-based optimization. International Joint Conference on Artificial Intelligence (IJCAI-20); DEMO track, 2020.

11 Xavier Gillard, Coppé Vianney, Pierre Schaus, and Ciré André. Improving the filtering of branch-and-bound mdd solvers. In *CPAIOR*, 2021.

12 D. Guo. Regionalization with dynamically constrained agglomerative clustering and partitioning (REDCAP). *International Journal of Geographical Information Science*, 22(7):801–823,

July 2008. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/13658810701674970. doi:10.1080/13658810701674970.

13 Paul A Longley, Michael F Goodchild, David J Maguire, and David W Rhind. *Geographic information science and systems*. John Wiley & Sons, 2015.

14 Assuncao Martins, Marcos Neves, Gilberto Câmara, and Domingos Da Costa Freitas. Efficient Regionalization Techniques for Socio-Economic Geographical Units Using Minimum Spanning Trees. *International Journal of Geographical Information Science*, 20:797–811, August 2006. doi:10.1080/13658810600665111.

15 James M Omernik, Shannen S Chapman, Richard A Lillie, Robert T Dumke, et al. Ecoregions of wisconsin. *Transactions of the Wisconsin Academy of Sciences, Arts and Letters*, 88:77–103, 2000.

16 Stan Openshaw. A regionalisation program for large data sets. *Computer Applications*, 3(4):136–147, 1973.

17 Stan Openshaw. Classifying and regionalizing census data. *Census users' handbook*, pages 239–270, 1995.

18 Ran Wei, Sergio Rey, and Elijah Knaap. Efficient regionalization for spatially explicit neighborhood delineation. *International Journal of Geographical Information Science*, 35(1):135–151, 2021. doi:10.1080/13658816.2020.1759806.