Uma imagem com Tipo de letra, logótipo, símbolo, Gráficos

Descrição gerada automaticamente

Uma imagem com Saturação de cores, Lilás

Descrição gerada automaticamente

**Comparing Similarity between two Territory Partitions in Political Districting Problems**

**Indices and practical issues**

**Sene Conté**

Thesis to obtain the Master of Science Degree in

**Computer Science and Engineering**

Supervisors: Prof. Luís Manuel Silveira Russo Prof. José Rui De Matos Figueira

**Examination Committee**

Chairperson: Prof. Pedro Tiago Gonçalves Monteiro

Supervisor: Prof. Luís Manuel Silveira Russo

Member of the Committee: Prof. Vasco Miguel Gomes Nunes Manquinho

**(Month Year)**

**Declaration**

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

**Abstract**

This study addresses the complex problem of assessing similarity between two partitions of a district map, where each partition is modeled as a connected, undirected, and planar graph. Comparing similarity between such partitions is crucial in applications like urban planning, political districting, and geographic analysis, where variations in partitioning can have significant impacts on representation and resource distribution. Traditional similarity indices, such as the Rand Index, Fowlkes-Mallows Index, and Jaccard Index, are used as foundational measures. However, these indices are extended to incorporate weighting factors for districts, allowing for a more accurate comparison by accounting for variations in district characteristics, such as population density or geographic importance. By using weighted similarity indices, this research aims to achieve a refined measurement of similarity that captures the nuanced impact of each zone. This approach offers an enhanced framework for partition analysis, improving upon existing techniques that often overlook regional disparities. Ultimately, the study presents a comprehensive solution for similarity assessment in partitioned maps, applicable to various fields that rely on accurate and equitable territorial analysis.

**Keywords**

Districting; Weighted Pair Counting; Similarity Indices; Urban Planning; Territorial Maps; Perturbations

**Resumo**

Este estudo aborda o problema complexo de avaliar a semelhança entre duas partições de um mapa distrital, onde cada partição é modelada como um grafo conectado, não-direcionado e planar. Comparar a semelhança entre estas partições é crucial em áreas como o planeamento urbano, a demarcação de distritos eleitorais e a análise geográfica, onde diferentes partições podem ter impactos significativos na representação e na distribuição de recursos. Utilizam-se índices de semelhança tradicionais, como o Índice de Rand, o Índice Fowlkes-Mallows e o Índice de Jaccard, como medidas fundamentais. No entanto, estes índices foram estendidos para incorporar fatores de peso dos distritos, permitindo uma comparação mais precisa ao considerar variações nas características dos distritos, como densidade populacional ou importância geográfica. Através do uso de índices de semelhança pesadas, esta investigação visa obter uma medição refinada da semelhança, capturando o impacto específico de cada zona. Esta abordagem oferece uma estrutura melhorada para a análise de partições, superando técnicas existentes que frequentemente ignoram as disparidades regionais. Por fim, o estudo apresenta uma solução abrangente para a avaliação de semelhança em mapas particionados, aplicável a diversas áreas que dependem de uma análise territorial precisa e equitativa.

**Palavras Chave**

Distritos; Contagem de Pares Pesadas; Índices de Similaridade; Planeamento Urbano; Mapas Territoriais; Perturbações.

Contents

[1 13](#_Toc185091755)

[Introduction 13](#_Toc185091756)

[**1.1 Political Districting** 13](#_Toc185091757)

[**1.2 Problem Modeling** 15](#_Toc185091758)

[**1.3 Motivation** 17](#_Toc185091759)

[**1.4 Organization of the Document** 18](#_Toc185091760)

[2 20](#_Toc185091761)

[Pair Counting Indices 20](#_Toc185091762)

[**2.1 Rand Index** 22](#_Toc185091763)

[**2.2 Fowlkes Mallows Index** 23](#_Toc185091764)

[**2.3 Jaccard Index** 24](#_Toc185091765)

[**2.4 Analysis of Index Properties** 25](#_Toc185091766)

[**2.4.1 Range Maximal and Minimal Agreement** 26](#_Toc185091767)

[**2.4.2 Symmetry** 26](#_Toc185091768)

[**2.4.3 Linear Complexity** 27](#_Toc185091769)

[**2.4.4 Distance** 27](#_Toc185091770)

[**2.4.5 Monotonicity** 28](#_Toc185091771)

[3 29](#_Toc185091772)

[Weighted Pair Counting 29](#_Toc185091773)

[**3.1 Mathematical Formulation** 29](#_Toc185091774)

[**3.2 Properties and Validations** 32](#_Toc185091775)

[4 34](#_Toc185091776)

[Implementation 34](#_Toc185091777)

[**4.1 Algorithm** 34](#_Toc185091778)

[**4.1.1 Key Inputs** 34](#_Toc185091779)

[**4.1.2 Key Outputs** 34](#_Toc185091780)

[**4.1.3 Algorithm Description** 35](#_Toc185091781)

[**4.1.4 Pseudocode** 36](#_Toc185091782)

[**4.2 Data structures** 37](#_Toc185091783)

[**4.3 Perturbation algorithm** 39](#_Toc185091784)

[**4.4 Complexity Analysis of Algorithms** 41](#_Toc185091785)

[5 44](#_Toc185091786)

[Experimental Results 44](#_Toc185091787)

[**5.1 Data Set Generation and Preprocessing** 44](#_Toc185091788)

[**5.2 Results** 44](#_Toc185091789)

[6 45](#_Toc185091790)

[Conclusion 45](#_Toc185091791)

[**6.1 Conclusions** 45](#_Toc185091792)

[**6.2 System Limitation and Future Work** 45](#_Toc185091793)

[7 Bibliography 46](#_Toc185091794)

# 

# Introduction

## **Political Districting**

Political districting is the complex task of dividing a region into distinct territorial units or districts, usually for electoral, administrative, or service provision purposes. The goal is to create balanced, representative districts that satisfy certain criteria, such as equal population, compactness, contiguity, and, importantly, a fair distribution that prevents gerrymandering. Despite these objectives, achieving optimal districting is computationally challenging, especially for large-scale regions, which often leads researchers to employ heuristic methods like Local Search (LS) and Simulated Annealing (SA) to find feasible, if not exact, solutions. These challenges underscore the critical need for well-defined districting approaches, as highlighted in Silva’s work on integrality and contiguity in political maps​​ [1].

Districting challenges are inherently multi-criteria, aiming to create “homogeneous” zones that satisfy demographic, geographical, and sometimes even ecological constraints. For example, districting applications range widely, from defining electoral boundaries to organizing police patrol areas or public service zones. Each of these cases involves grouping elementary units into contiguous clusters that reflect a balanced mix of attributes, thus forming a range of viable solutions that often represent a compromise between conflicting objectives. The search for optimal solutions in such scenarios is generally replaced with an effort to find non-dominated solutions, which are judged based on criteria like homogeneity, geographical continuity, and other region-specific constraints​ [2].

In police districting, the objective is to design patrol sectors that distribute workload evenly across areas, considering factors like response times and operational efficiency. Initially, police sectors were manually drawn based on visible geographical features, but modern approaches incorporate Geographic Information Systems (GIS) and predictive analytics to optimize workload balance and patrol efficiency. Recent research emphasizes the importance of automation and the integration of data-driven methods, which can help create well-balanced, contiguous patrol zones and reduce response times across sectors​ [3].

Political districting’s challenges and solutions reflect broader socio-political implications, as equitable district boundaries directly impact democratic representation, service accessibility, and public trust in governance. Advances in mathematical modeling and computational heuristics, such as evolutionary algorithms and local search methods, continue to shape this field, helping decision-makers explore a diverse array of districting solutions that can balance population distribution with geographical constraints. These tools play a significant role in making districting an increasingly transparent and representative process across applications.

A critical aspect of political districting, particularly when revising or comparing existing maps, is the comparison of alternative partitions. Comparing two territory partitions is essential for evaluating changes in district boundaries, assessing their alignment with demographic shifts, and ensuring that new configurations meet fairness and representativeness standards. Pereira et al. (2009) introduced indices for comparing partitions, including compatibility, inclusion, and distance measures. These indices provide a systematic way to assess how closely an alternative map resembles an existing one, offering insight into the differences between configurations that may have socio-political or operational implications. This method is particularly valuable in evaluating proposed political districting maps against established ones to ensure continuity or manage gradual transitions in boundary adjustments [4].

In this thesis, we focus on the comparative analysis of political districting solutions, specifically examining how two territorial partitions can be evaluated for similarity. By leveraging the methodologies discussed above, we aim to develop a robust framework for comparing political district configurations. This comparative approach will contribute to a deeper understanding of how district boundaries evolve and how new proposals impact existing social, political, and operational contexts.

## **Problem Modeling**

In political districting (PD), the challenge of partitioning a given territory into distinct, cohesive zones, or districts, is critical for ensuring fair representation and efficient public service provision. Each district, designed to meet demographic, geographic, and socio-economic criteria, must balance population, maintain geographical contiguity, and optimize compactness. Beyond initial partitioning, a significant aspect of PD involves comparing alternative district maps to assess similarity. Such comparisons are essential when proposing new maps, as they allow decision-makers to evaluate how well new districts align with or diverge from existing configurations, taking into account both demographic and spatial factors.

To model this problem, we represent the territory as a contiguity graph , where is the set of vertices, each representing an indivisible elementary unit of territory (such as a municipality), and is the set of edges, where an edge represents adjacency between two units and . This graph is structured to be connected, undirected, and planar, which ensures that the entire territory forms a single contiguous entity, that each adjacency is reciprocal, and that it can be represented without overlapping edges, reflecting realistic territorial boundaries.

Within this framework, a **zone** is defined as a subset of contiguous elementary units, while a **partition**  is a collection of zones that collectively cover the territory without overlap, forming a cohesive district map. In comparing two different partitions, we denote one as and the other as .

For each elementary unit there is one and only one zone such that belongs to . For the sake of simplicity, an elementary unit, is also represented by its index . *Figure 1*(a) represents a territory composed of 16 elementary units, divided into four zones, . *Figure 1*(b) shows the contiguity graph G corresponding to the territory of *Figure 1*(a). To facilitate meaningful comparisons, we assign characteristic values to each elementary unit based on a specific property (e.g., population, area, or economic output). For each unit , the **characteristic function** returns the value of a chosen attribute. Using this, we define the **zone sum ,** representing the total characteristic value across all units in a zone , as . The **total characteristic** for the entire territory is then , enabling us to calculate the **zone weight .** This weight provides a basis for weighted comparisons, allowing zones with higher demographic or economic significance to contribute proportionally to the similarity assessment.

The following definitions establish the basis for comparing two partitions in terms of their zones and their constituent units. First, an **attribute** in a contiguity graph is a real-valued function defined on with non-negative values. For any subset , the characteristic sum denotes the overall attribute value for that subset, and is the overall attribute for as a whole. This function is fundamental for evaluating and comparing the similarity of characteristics between zones and partitions.

Two zones and are considered **equal** with respect to an attribute if each is included within the other for the attribute under consideration, denoted . Similarly, two partitions and are considered **equal** with respect to the attribute if, for every pair of zones , either or the two zones do not overlap in terms of the characteristic. This definition of equality ensures that both partitions are equivalent in terms of their overall structure and the distribution of the attribute across zones.

A grid of circles and numbers

Description automatically generated with medium confidence

Figure 1. A territory and the associated contiguity graph.

A pair of squares with lines and letters

Description automatically generated with medium confidence

Figure 2. Four different partitions, Y, Y', Y'', and Y’’’.

## **Motivation**

In the realm of political districting, ensuring fair and representative territorial partitions is critical to democratic processes, public administration, and resource allocation. The need to compare and evaluate territorial partitions arises frequently, particularly in scenarios involving electoral redistricting, urban planning, or public service zoning. A partition comparison allows decision-makers to assess the extent to which a proposed or alternative configuration aligns with historical or current partitions. This is particularly important in preserving continuity, minimizing disruptions, and ensuring equity across districts. The research of Tavares Pereira et al. emphasizes the socio-economic importance of comparing partitions to address discrepancies and optimize decision-making frameworks [4].

The challenge of comparing partitions extends beyond political applications. The Adjusted Rand Index (ARI) and other pair-counting methods have become standard tools in cluster analysis and unsupervised machine learning, where they serve as benchmarks for validating clustering algorithms. These indices provide insights into the alignment of clusters across diverse applications, from biology to computational linguistics​ [5]. In particular, the evolution of indices to accommodate asymmetric and weighted scenarios reflects the growing complexity of real-world partition comparison problems.

Weighted comparisons are indispensable in many fields where certain attributes carry more significance than others. In political districting, for instance, population balance and geographical contiguity often take precedence over compactness. Similarly, in sports, weight measures are used to rank players and teams. For instance, let’s take an example of European Football, The European Golden Shoe, also known as European Golden Boot, is an award that is presented each season to the leading goal scorer in league matches from the top division of European national leagues. It has been calculated using a weighting in favor of the highest ranked leagues. Between 1968 and 1991, the award was given to the highest goal scorer in any European league, regardless of the strength of the league in which they played. But following some incidents, since 1996-97 season, the award has been awarded based on a point system that allows players in tougher leagues to win even if they score fewer goals than a player in a weaker league. The weightings are determined by the league’s clubs in European competitions. Goals scored in the top five leagues (see [6]) according to the UEFA coefficient list are multiplied by factor of two, goals scored in the leagues ranked 6 to 22 are multiplied by factor of 1.5 and goals scored in the leagues ranked 22 and above are multiplied by factor of 1 [7].

The use of weights in neural networks are pivotal for learning, optimization, and decision-making. They determine the influence of each input feature on the model's predictions, effectively guiding the learning process. Han et al. (2015) highlighted the profound impact of learning both weights and connections for improving the efficiency of neural networks. Their work demonstrated that by pruning low-weight connections, those deemed less significant—the computational cost of neural networks can be reduced by orders of magnitude without compromising accuracy. For example, in the case of AlexNet, they achieved a ninefold reduction in parameters, making neural networks more accessible for deployment on resource-constrained devices like mobile systems. This illustrates the power of weights in maintaining a balance between performance and computational efficiency, a principle that resonates with the need for weighted analysis in comparing territorial partitions [8].

Beyond efficiency, weights also enable neural networks to address real-world challenges such as imbalanced data distributions. In class-imbalanced learning, the dynamically weighted loss functions proposed by Wei-Dong et al. ensure that underrepresented classes receive greater attention during training. This approach not only reduces bias but also enhances the network's confidence calibration, leading to fairer and more reliable predictions. This methodology underscores the adaptability and significance of weights in handling diverse scenarios, from image classification to complex societal problems​ [9].

These diverse applications illustrate the profound impact of weighted comparisons across domains. The ability to incorporate weights into pair-counting indices not only enriches the analysis but also makes it more adaptable to specific contexts and objectives. By extending and refining these tools for political districting, this research aims to provide a robust framework for comparing territorial partitions, addressing both the intrinsic challenges of districting and the broader applicability of weighted similarity measures.

## **Organization of the Document**

|  |  |
| --- | --- |
| **Notation** | **Description** |
|  | Territory, where each represents an indivisible elementary unit. |
|  | A zone, defined as a set of contiguous elementary units (e.g., municipalities). |
|  | A partition or district map of the territory |
| *γ* | A partition or district map of the territory |
|  | Characteristic function returning the value of a specific characteristic (e.g., population, area) for an elementary unit |
|  | Total characteristic value for a zone , calculated as |
|  | Total characteristic value for the entire territory , given by |
|  | Weight of a zone , calculated as . |
|  | Contiguity graph representing territory , where is the set of vertices (elementary units) and is the set of edges (borders between adjacent units). |
|  | Set of vertices in the contiguity graph, representing elementary units. |
| } | Set of edges in the contiguity graph, where represents a border between adjacent units and |

Table 1. Table of Concepts

# 

# Pair Counting Indices

Pair counting is a foundational approach used in comparing partitions of a dataset, particularly in clustering and partition analysis. It evaluates the relationships between pairs of elements, focusing on whether they are grouped similarly in two different partitions. By considering all possible pairs of elements, pair counting provides a robust basis for quantifying the similarity or dissimilarity between two partitions​ [10] [5].

Consider two partitions to be similar if they agree on many pairs. Formally, let , suppose and are two different partitions of territory , , for example, a reference standard partition and a trial partition that was obtained with a partitioning method that is being evaluated. Let denote the number of elementary units that are common to zone of the first partition and in zone of the second partition. Then the zone overlaps between the two partitions and can be written in the form of a contingency table where and are the number of elementary units in zones and respectively:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Zone |  |  | … |  | Sum |
|  |  |  | … |  |  |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
| . | . | . | . | . | . |
|  |  |  | … |  |  |
| Sum |  |  |  |  |  |

Table 2: Contingency table

Based on the contingency table which is also called a matching table or cross-classification table in the field of cluster analysis, some very important quantities can be obtained. These are the basis for many indices used for comparing two partitions based on counting pairs and are often presented in the form of a mismatch matrix [10, 11, 12]:

|  |  |  |
| --- | --- | --- |
|  | Pair in same zone in | Pair in different zone in |
| Pair in same zone in |  |  |
| Pair in different zone in |  |  |

Table 3: Describing zone assignment of pairs of observations between two partitions and

Where the quantities , , , and are sometimes denoted differently. These quantities are defined as follows:

* Elementary units in the pair that are placed in the same zone in and in the same zone in ′

* Elementary units in the pair that are placed in the same zone in and in different zone in ′

* Elementary units in the pair that are placed in different zone in and in the same zone in ′

* Elementary units in the pair that are placed in different zone in and in different zone in ′

**Definition 1**. A pair counting index is a similarity index that can be expressed as a function of the pair counts , , , and .

In the literature, more than 27 pair counting similarity indices have been introduced. The indices are shown in *table 4*. However, this document focuses on three of them: the Rand Index, the Fowlkes-Mallows index, and the Jaccard index.

## **Rand Index**

The Rand Index () is a widely used measure for quantifying the agreement between two partitions of a dataset. Introduced by Rand in 1971, it evaluates the similarity of two partitions by assessing pairwise relationships among elements within the dataset. Specifically, the index considers all possible pairs of elementary units and evaluates whether each pair is consistently classified in the same or different zones across the two partitions. This simple yet robust approach makes the a cornerstone metric for external validation in partition comparison tasks​​ [13, 14].

The is a member of the family of pair-counting measures, focusing on pairs of elementary units and their zones assignments. It quantifies similarity by comparing the number of concordant and discordant pairs between two partitions​ [15] .

Based on the contingency table **,** the is defined as:

The denominator represents the total number of pairs of objects in the dataset, which is equal to , where is the number of elementary units. The numerator measures the number of pairs where the two partitions agree, either because they zone the elementary units together or separately .

The ranges from 0 to 1, where higher values indicate greater similarity between the two partitions. A value of 1 signifies perfect agreement, meaning the partitions are identical in how they group the elementary units. In contrast, a value of 0 indicates no agreement, suggesting that the partitioning decisions are entirely independent and do not align in any meaningful way. This range provides an intuitive scale for evaluating the degree of correspondence between partitions.

The assesses both agreements (pairs grouped together or apart in both partitions) and disagreements (pairs grouped differently in the two partitions). By considering both and , the index captures a holistic view of the partition alignment. However, the does not penalize chance agreement, which can inflate its value when comparing partitions with many zones. This limitation led to the development of adjusted indices like the Adjusted Rand Index, which accounts for random chance​ [16].

The serves as a foundational component for similarity measurement in this thesis, offering a pair-counting perspective that is both intuitive and adaptable. While the unadjusted Rand Index provides baseline insights, integrating weighted pair-counting methods into the analysis enriches its applicability, particularly in scenarios where certain demographic or geographic attributes are prioritized. This adaptation aligns the metric more closely with the nuanced challenges of political districting, enhancing its utility as a tool for evaluating and comparing partitions in complex real-world settings​.

## **Fowlkes Mallows Index**

The Fowlkes-Mallows Index () is a statistical measure designed to assess the similarity between two clustering results. Introduced as a method for validating clustering quality, the bridges the gap between precision and recall by leveraging their geometric mean. Its primary strength lies in its ability to evaluate pairwise similarity between elements, making it particularly useful for applications where both true positive and false positive rates play crucial roles in assessing clustering performance. This property makes the highly applicable to problems of partition comparison, such as those encountered in political districting​.

Mathematically, the Fowlkes-Mallows Index is defined as:

The numerator, , represents the count of correctly clustered pairs, while the two fractions evaluate precision and recall , respectively. The is thus inherently tied to both the accuracy and completeness of the partitioning alignment.

The ranges between 0 and 1. A value closer to 1 indicates high similarity between the partitioning results, implying that most pairs of elements are correctly assigned to the same or different zones. Conversely, a value closer to 0 suggests a lack of agreement, highlighting discrepancies between the partitions. Unlike some other indices, the FMI assumes the availability of ground truth or benchmark classifications for comparison, which positions it as a reliable tool for externally validating partitioning solutions​.

In the context of political districting, the becomes a valuable metric for evaluating the similarity between territorial partitions. By treating districts as clusters and elementary units as data points, the can assess how well two districting plans align in terms of pairwise assignments. For instance, when comparing proposed and historical districting plans, the offers insights into the extent of continuity or change in territorial configurations. Furthermore, the incorporation of weights—based on demographic or geographic significance—extends the ’s utility, enabling nuanced assessments that account for real-world constraints and priorities​ [17, 18, 19].

The ’s reliance on pairwise comparisons also aligns with other similarity measures, such as the Rand Index, but its focus on precision and recall integration provides a distinct perspective. This thesis explores its application in districting analysis, demonstrating its potential to inform decisions where fairness, contiguity, and demographic representation are paramount. By adapting the to weighted contexts, this research seeks to enhance its applicability to complex real-world scenarios, making it a robust tool for partition comparison in political and other spatially constrained domains​.

## **Jaccard Index**

The Jaccard Index (), also referred to as the Jaccard similarity coefficient, is a widely recognized metric for measuring the similarity between two sets or partitions. It was initially introduced by Paul Jaccard in 1901 to quantify the similarity between ecological species compositions but has since been adapted for numerous applications across various fields, including clustering, image segmentation, and partition comparison. In the context of political districting, the provides a robust means to evaluate the degree of similarity between two territorial partitions by focusing on shared and distinct elements between them​ [20].

Formally, the is defined as the size of the intersection divided by the size of the union of the sets under comparison. For the purpose of partition comparison in this thesis, the formula is adapted to pair-counting components as follows:

The numerator captures the shared relationships between the two partitions, while the denominator encompasses all pairwise relationships, including those that diverge. Notably, , which represents pairs that are not grouped together in either partition, is excluded from the formula, as the focuses solely on positive pairwise interactions.

The ranges between 0 and 1, where 1 signifies perfect similarity (i.e., both partitions cluster every pair identically), and 0 indicates no similarity (i.e., no overlap in pairwise clustering between the partitions). Its focus on shared and divergent pairwise relationships makes it particularly well-suited for contexts where the presence of common elements is more informative than their absence**​** [21, 22]**.**

In the domain of political districting, the offers an intuitive and straightforward means to evaluate and compare district maps. By considering zones or districts as clusters and their constituent elements as data points, this index provides insights into the continuity or changes between proposed and existing district maps. For instance, it can measure how well a redistricting plan preserves neighborhood integrity or reflects demographic cohesion. Incorporating demographic or geographic weights into the calculation further extends its applicability, allowing policymakers to account for the relative importance of different characteristics in their evaluations [20].

The is also frequently compared with other similarity measures, such as the Rand Index, due to its sensitivity to shared relationships and exclusion of irrelevant pairs (i.e., ). This characteristic makes it particularly advantageous in scenarios with sparse overlap between partitions. While the index does not explicitly address chance similarity, it serves as a valuable complement to adjusted indices, which adjust for expected overlap. Its use in this thesis underscores its flexibility and relevance in analyzing and comparing complex spatial and demographic partitions.

## **Analysis of Index Properties**

Evidently, many different partition similarity indices are used by researchers and practitioners. A natural question is: *how to choose the best one?* Before trying to answer this question, it is important to understand whether the problem is relevant, indeed, if the indices are very similar to each other and agree in most practical applications, then one can safely take any index. In this section, we motivate and formally define properties that are desirable for our partition similarity indices. Then, we give the proofs for pair counting similarity indices mentioned above. For such indices, we interchangeably use the notation and .

### **Range Maximal and Minimal Agreement**

* **Maximal Agreement:** The numerical value that an index assigns to similarity must be easily interpretable. It should be easy to see whether the candidate partition is maximally similar to (i.e, coincides with) the reference partition. Formally, we require that is constant and is a strict upper bound for for all .

**Definition 2.** A pair counting index has maximum agreement property if there exists a constant so that with equality if and only if .

For Rand Index, Fowlkes Mallows Index and Jaccard Index . This property is easy to check, and it is satisfied by Rand Index, Fowlkes Mallows Index and Jaccard Index. Can trivially be tested by simply checking .

* **Minimal Agreement:** The maximal agreement property makes the upper range of the index interpretable. Similarly, a numerical value for low agreement would make the lower range interpretable. A minimal agreement is not well defined for general partitions: it is unclear which partition is most dissimilar to a given one. However, pair-counting indices form a subclass of graph similarity indices. For a graph with edge set E, it is clear that the most dissimilar graph is its complement (i.e., with edge-set ) [23]. Comparing graphs to complement results in pair-counts and . This motivates the following definition:

**Definition 3**. A pair counting index has the minimal agreement property if there exists a constant so that with the equality if and only if .

It is easy to verify that this property is satisfied by Rand Index, Fowlkes Mallows and Jaccard Index by simply checking .

### **Symmetry**

Similarity is intuitively understood as a symmetric concept. Therefore, a good similarity index is expected to be symmetric, i.e., for all partitions and . Most of the pair counting indices are symmetric, including Rand Index, Fowlkes Mallows Index and Jaccard Index.

### **Linear Complexity**

For comparing two large partitions, running time is crucial, and algorithms with superlinear time can be infeasible. In these cases, a validation index with super linear running time would be a significant bottleneck. Furthermore, computationally heavy indices also tend to be complicated and hard to interpret intuitively. We say that an index has linear complexity when its worst-case running time is . In this context, denotes the number of elementary units within the partitions. All pair counting similarity indices satisfy this property.

**Lemma 1**. The nonzero values of can be computed in expected time O(n).

**Proof**. We will store these nonzero values in a hash-table (Figure 1) that maps the pairs . These values are obtained by iterating through all elements and incrementing the corresponding value of . For hash-tables, searches and insertions are known to have amortized complexity expected time , meaning that any sequence of 𝑛 such actions have worst-case running time of, from which the result follows.

### **Distance**

For some applications, a distance-interpretation of dissimilarity may be desirable: whenever is similar to  and is similar to , then should also be somewhat similar to . For example, assume that the reference partition (e.g., labeled by a partitioning algorithm) is an approximation of the root partition. In such situations, it may be reasonable to argue that the reference partition is at most a distance Ɛ from the root partition so that triangle inequality bounds the dissimilarity of the candidates partition to the unknown root partition.

A function is a distance metric if it satisfies three distance axioms: 1) symmetry 2) positive definiteness ; 3) the triangle inequality We say that is linearly transformable to a distance metric if there exists a linearly equivalent index that satisfies these three distance axioms.

Note that all three axioms are invariant under rescaling of . We have already imposed symmetry as a separate property, and positive definiteness is equivalent to the maximal agreement property. Therefore, whenever V has these two properties, it satisfies the distance property iff satisfies the triangle inequality, for

**Rand Index**. The Mirken metric corresponds to a rescale version of the size of the symmetric difference between the sets of intra-cluster pairs. The symmetric difference is known to be a distance metric.

**Jaccard Index**: in [24], it is proven that the Jaccard distance is indeed a distance metric.

**Fowlkes Mallows Index**. In the previous section, we prove that satisfies symmetry and . Therefore, the distance also satisfies positive definiteness. The only property we need to prove now is triangle inequality. If we transform distance values into similarity values within the range between using function , we obtain for property triangle inequality:

. And thus,

holds (see [25]).

Since Fowlkes Mallows distance holds triangular inequality, then is a distance metric.

### **Monotonicity**

When one partition is changed such that it resembles the other partition more, the similarity score ought to improve. Hence, we require an index to be monotone w.r.t. changes that increase the similarity. This can be formalized via the following definition:

**Definition 4**. For partition and , we say that is an consistent improvement of iff and all pairs of elements agreeing in and also agree in and .

This leads to the following monotonicity property:

**and** . It can easily be seen that these indices are increasing in while decreasing in For we note that whenever gets increased, either or must decrease, resulting in an increase of the index. Therefore, these indices satisfy monotonicity.

# 

# Weighted Pair Counting

Weighted pair counting is an innovative extension of traditional pair counting methods, designed to address scenarios where the significance of individual zones in a partition varies based on their characteristics. This approach introduces weights to account for attributes like population, geographic size, or socio-economic factors, providing a more nuanced and realistic evaluation of partition similarity. By incorporating weights, this method ensures that zones with higher importance exert a proportional influence on the comparison metrics, aligning the analysis with real-world priorities.

In weighted pair counting, the weight of each zone lies within the interval and the sum of weights across all zones equals 1. This normalization ensures that the weighted components retain the desirable properties of traditional pair counting indices, such as linear complexity, symmetry, and monotonicity.

## **Mathematical Formulation**

To formalize weighted pair counting, we redefine the traditional pair counting components and to include zone weights. These components quantify the relationships between pairs of elements within and across zones in two partitions and , where zones in are assigned weights and zones in are assigned weights

* **: Weighted pairs in the same zones in both partitions**

This component counts the weighted number of pairs of elements that belong to the same zone in both partitions.

Where:

* : is the number of elements common in zone and in the first and second partitions respectively.
* : is the weight of zone in the first partition.
* : is the weight of zone in the second partition.
* **: Weighted pairs in the same zone in but different zones in**

This component captures the weighted pairs grouped together in a single zone in but scattered across different zones in .

Where:

* : is the number of elements common in zone and in the first and second partitions respectively.
* : is the number of elements in zone in the first partition.
* : represents the sum of the weights of zones in the second partition where the number of elements shared with the first partition, 𝑛𝑖𝑗, is greater than 2.
* **: Weighted pairs in different zones in but the same zone in**

This component quantifies the weighted pairs grouped together in a single zone in but distributed across multiple zones in .

Where:

* : is the number of elements in zone in the second partition.
* : represents the sum of the weights of zones in the first partition where the number of elements shared with the second partition, , is greater than 2.
* **: Weighted pairs in different zones in both partitions**

This component counts the weighted number of pairs of elements that belong to different zones in both partitions.

Where:

* : is the number of elementary units in the partitions

By incorporating zones weights into the computation of pair counting components, it is possible to adjust the contribution of each pair of elements based on the characteristics of zones to which they belong. This allows for a more nuanced approach of partition similarity that considers the distribution of elements within zones and the relative importance of different zones.

## **Properties and Validations**

Weighted pair counting retains the core properties of traditional pair counting indices, ensuring consistent and interpretable similarity measurements. The introduction of weights does not alter the fundamental relationships between the components, as the total number of pairs remains invariant:

Let’s demonstrate that the sum of weighted components, assuming the weights are normalized is equal to

After simplification, we get:

Given that the weights are normalized , the expression simplifies further:

This ensures compatibility with existing pair counting similarity indices, allowing for seamless adaptation to weighted scenarios. By prioritizing zones with greater significance, weighted pair counting provides a more equitable and context-sensitive approach to comparing partitions, particularly in complex applications like political districting.

# 

# Implementation

The implementation of this research focuses on a structured approach to comparing similarity between two territorial partitions in the context of political districting. The process integrates several computational techniques, including graph-based modeling, perturbation algorithms, and similarity indices, to evaluate how closely two partitions align. This section outlines the algorithms, data structures, perturbation methods, complexity considerations, and the overall solution that forms the basis of the implementation.

## **Algorithm**

The core algorithm for comparing the similarity between two territorial partitions operates on the principle of pair counting, leveraging contingency tables and similarity indices to measure alignment. This method integrates traditional and weighted pair counting components, ensuring a nuanced analysis of partition similarity.

### **Key Inputs**

1. Root Partition Graph : The original partition represented as a graph where vertices are elementary units and edges define adjacency.
2. Comparing Partition Graph : The partition being compared initially identical to , but subject to perturbation during execution.
3. Number of Perturbation Iterations : The number of iterations to perturb .
4. Similarity Index : The index to compute the similarity, selected from Rand Index, Fowlkes-Mallows Index, or Jaccard Index.

### **Key Outputs**

* **Similarity Value**: A numerical measure of similarity between the two partitions based on the selected index.

### **Algorithm Description**

1. **Initialization**:
   * Load the root partition graph and initialize the comparing partition graph .
2. **Perturbation**:
   * Apply perturbation iterations to . Each perturbation modifies the partitioning structure, creating variations for comparison.
3. **Zone and Unit Identification**:
   * For each zone in and , retrieve the identifiers of all elementary units (vertices) contained in the zone.
4. **Weight Update:** 
   * Update the weights for each zone. The weight of a zone is calculated based on the attributes of its constituent elementary units, with each unit contributing to its pre-defined weight.
5. **Contingency Table Calculation:** 
   * Construct the contingency table using the identifiers of elementary units from both partitions. The table represents the number of elementary units shared between zones in and .
6. **Pair Counting Components Calculation**:
   * Using the contingency table, compute the traditional pair counting components .
   * Extend the computation to weighted components, incorporating the updated zone weights.
7. **Similarity Computation:**
   * Using the chosen similarity index , calculate the similarity between and . The similarity value is derived from the pair counting components and is returned as the final output.

### **Pseudocode**

**Algorithm 1.** Partition Comparison and Similarity Evaluation

**Require:** is an undirected, connected, and planar

**Input**:

root partition graph, representing the original partition

comparing partition graph

number of perturbation iterations applied to

similarity index used to compute the similarity (

**Output**:

SimilarityValue The computed similarity measure

1. Initialize =

2. **for** r = 1 to **do**

3. Perturb

4. **end for**

5. **for** each zone in and **do**

6. Retrieve identifiers of elementary units

7. Update weights for each zone

8. **end for**

9. ContingencyTable ← ComputeContingencyTable(, )

10. () ← ComputePairCountingComponents(ContingencyTable)

11. WeightedComponents ← ComputeWeightedComponents(ContingencyTable)

12. SimilarityValue ← ComputeSimilarityIndex(, WeightedComponents, )

13. **return** SimilarityValue

This algorithm provides a structured approach to evaluate partition similarity, integrating both traditional and weighted methods. The perturbation and weighting steps introduce variability and depth, enabling nuanced insights into how partitions align in different scenarios.

## **Data structures**

The implementation of the algorithm relies on a carefully designed set of data structures to model the essential components of the political districting problem. These data structures represent zones, elementary units, and their relationships, enabling efficient partition representation, computation, and manipulation. The foundational elements of the data structures are the **ZoneNode** and **ElementaryUnit Node** classes, which encapsulate the characteristics and interactions of zones and their constituent elements. Additionally, the graph is modeled as a collection of ZoneNodes, stored in a vector for simplicity and scalability.

A ZoneNode is represented by the following data structure:

**Structure** ZoneNode

*{*

*Id*

*elementaryUnits*

*boundaryVertices*

*weight*

*}*

A **ZoneNode** represents an individual zone in the partition, encapsulating information about its identity, constituent elementary units, boundary vertices, and weight. The ***id*** attribute uniquely identifies the zone, allowing for clear differentiation between zones during computations and manipulations. The ***elementaryUnits*** attribute is a set of pointers to the elementary units that belong to the zone. This set provides quick access to the elements contained within the zone, which is crucial for operations like weight calculations and similarity comparisons.

To enhance the efficiency of the perturbation algorithm, the ***boundaryVertices*** attribute stores a vector of pointers to elementary units located on the boundary of the zone. Boundary vertices are elementary units that share edges with vertices in neighboring zones. By maintaining a list of these vertices, the algorithm can efficiently identify potential candidates for movement between zones, reducing unnecessary computations. The ***weight*** attribute stores the total weight of the zone, derived from the weights of its constituent elementary units. This attribute is central to the weighted pair counting methodology, as it determines the influence of the zone in similarity calculations.

An Elementary Unit Node is represented by the following data structure:

**Structure** ElementaryUnit

*{*

*Id*

*char*

*neighbors*

*parentZone*

*NeighborZones*

*}*

An **ElementaryUnitNode** represents a single elementary unit or vertex within the graph. Each elementary unit is uniquely identified by its ***id***, ensuring clear differentiation across the graph. The ***char*** attribute represents the characteristic value of the elementary unit. This value contributes to the calculation of the zone's weight, as it reflects a measurable property of the unit, such as population or area.

The ***neighbors*** attribute is a set of pointers to adjacent elementary units, effectively modeling the adjacency relationships of the graph. This representation enables efficient traversal and connectivity checks, which are essential for ensuring contiguity in zone configurations. The ***parentZone*** attribute maintains a pointer to the zone to which the elementary unit currently belongs. This linkage establishes a direct relationship between elementary units and their parent zones, simplifying operations like weight updates and partition comparisons.

To further optimize the perturbation algorithm, the ***neighborZones*** attribute is included. This vector contains pointers to neighboring zones, defined as zones that share at least one edge with the current elementary unit. By precomputing and maintaining a list of neighboring zones, the algorithm can efficiently identify target zones for transferring boundary vertices during perturbation, streamlining the process.

The **Graph** structure is implemented as a vector of ZoneNode pointers. This representation provides a scalable and dynamic means of storing and managing zones within the partition. Each ZoneNode encapsulates the elementary units it contains, their boundaries, and their relationships, ensuring that the graph structure is both comprehensive and efficient for the operations required by the algorithm.

This data structure design balances flexibility, efficiency, and scalability, enabling the algorithm to handle complex partition comparisons effectively. By encapsulating the relationships and characteristics of zones and elementary units, these structures provide a robust foundation for the implementation and ensure that operations like perturbation, weight updates, and similarity calculations are performed with precision and efficiency.

## **Perturbation algorithm**

Perturbation is a foundational technique for exploring the solution space in optimization problems. It involves introducing small, deliberate changes to a given configuration to evaluate and improve the solution. In clustering problems, perturbation plays a significant role in algorithms like Simulated Annealing (SA), which iteratively generates and evaluates neighboring solutions to optimize an objective function. The **Simulated Annealing Clustering Algorithm Based on Center Perturbation Using Gaussian Mutation** demonstrates how perturbation can alter cluster centers to achieve improved solutions [26]. Similarly, the **Research on Improved Genetic Simulated Annealing Algorithm** highlights the use of perturbation in multi-objective optimization, demonstrating its effectiveness in balancing multiple objectives while maintaining solution diversity​ [27].

In the context of this thesis, perturbation is adapted to the political districting problem, ensuring the structural properties of the graph—connectedness, planarity, and undirected edges—are preserved. The perturbation algorithm focuses on efficiently modifying territorial partitions while maintaining their validity as spatial configurations.

The perturbation algorithm introduced in this work builds upon these concepts, tailoring them to the districting problem. The algorithm is designed to modify the partition structure iteratively, facilitating exploration of the solution space to compare similarity between partitions. The key steps of the algorithm, as detailed in the proposal are:

* **Random Zone Selection**: The algorithm begins by selecting a zone at random from the list of available zones in the graph. This zone will serve as the source of perturbation.
* **Boundary Identification**: For each zone, the algorithm identifies boundary vertices—elementary units adjacent to units in other zones. This step leverages the data structures defined earlier, where boundary vertices are stored for efficient access.
* **Selection of Boundary Vertex**: A random boundary vertex is selected. The selection of boundary vertices ensures that the perturbation affects only the edges between zones, preserving the overall structure.
* **Move the Boundary Vertex:** The selected boundary vertex is then reassigned to one of its neighboring zones. This involves:
  + Removing the vertex from its current zone.
  + Adding the vertex to the neighboring zone
  + Updating the internal representation of the graph to reflect this movement.
* **Update Boundary Vertices:** The process of updating boundary vertices is a critical aspect of the perturbation algorithm, ensuring that boundary elements of zones are correctly identified and maintained. A boundary vertex is an elementary unit (vertex) that shares at least one edge with a vertex in a neighboring zone. Accurate identification and updating of these vertices are essential for efficient and valid perturbations during the algorithm's execution.

To update the boundary vertices of a zone, the algorithm iterates over all elementary units within the zone. For each elementary unit, it inspects its neighbors to check if any are situated in a different zone. If a neighboring elementary unit belongs to a different zone, the algorithm designates the current elementary unit as a boundary vertex and adds it to the boundary vertices list of its parent zone. Simultaneously, it updates the list of neighboring zones for the current elementary unit, ensuring that the adjacency relationships between zones remain consistent.

**Algorithm 2.** Update Boundary Vertices

**Input**: ZoneNode zone

**Output**: Updated boundary vertices for the input zone

1. **zone.boundaryVertices.clear()** //Clear existing boundary vertices

2. **for each** vertex in zone.elementaryUnits **do**

3. **for** **each** neigh in vertex.neighbors **do**

4. **if** vertex.parentZone.id != neigh.parentZone.id **then**

5. vertex.parentZone.boundaryVertices.push\_back(vertex)

6. vertex.addNeighborZone(neigh.parentZone)

7. **end if**

8. **end for**

9. **end for**

* **Iteration**: The process repeats for a predefined number of iterations, exploring multiple configurations.

Preserving the graph’s structural properties is crucial. Connectedness ensures that each zone remains a cohesive unit, planarity guarantees the partition is realistic and adheres to geographic constraints, and undirected edges maintain the integrity of adjacency relationships between units. By incorporating these constraints into the perturbation algorithm, the resulting configurations remain valid for the comparison and analysis processes.

**Algorithm 3.** Algorithm Perturb Partition

**Require:** is an undirected, connected, and planar

**Ensure:** Modified Partition Graph remains undirected, connected and planar after perturbation.

**Input**:

PartitionGraph

NumberOfIterations R

**Output**:

Modified PartitionGraph

1. **for** i = 1 to **do**

2. Select a random zone from

3. Identify boundary vertices of

4. Select a random boundary vertex

5. Identify neighboring zones of

6. Randomly reassign to a neighboring zone

7. Update Boundary Vertices for and .

8. **end for**

9. **return**

By combining the principles of perturbation with domain-specific constraints, this algorithm effectively addresses the challenges of modifying territorial partitions of similarity analysis. Its design ensures that the process remains computationally efficient while generating meaningful variations for comparison. This approach not only builds on existing methodologies but also adapts them to the unique requirements of political districting.

Figure 3 illustrates an example of an undirected, connected, and planar graph partitioned into three distinct zones. Each zone contains a set of vertices representing elementary units, with certain vertices positioned at the boundary between zones to facilitate inter-zone connectivity. The figure highlights the critical role of boundary vertices in maintaining the structural integrity of the graph while enabling perturbation operations during partition comparison.

A group of circles with dots and lines

Description automatically generated

Figure 3: An undirected, connected, and planar graph partitioned into three zones.

Zone A consists of three vertices, two of which are boundary vertices, visually distinguished in red. These vertices establish adjacency with vertices from neighboring zones, marking them as key points of interaction between partitions. The remaining vertex within Zone A is internal, contributing to the overall contiguity of the zone but not directly influencing neighboring zones.

Zone B, which contains four vertices, features one boundary vertex, represented in green. This vertex serves as the sole point of connection between Zone B and adjacent zones, emphasizing the importance of even a single boundary vertex in preserving inter-zone relationships. The three internal vertices in Zone B contribute to the structural stability of the zone without extending their influence to neighboring zones.

Zone C is composed of three vertices, with its boundary vertices represented in blue. These vertices provide the necessary link between Zone C and adjacent zones, reinforcing the interconnected nature of the graph. The remaining vertices within Zone C, shown in black along with the internal vertices from Zones A and B, collectively maintain the contiguity and planarity of the graph.

The use of distinct colors for boundary vertices enhances the clarity of the figure, making it easier to visualize the interaction between zones. This visual representation underscores the significance of boundary vertices in graph perturbation algorithms, where the movement of such vertices can alter the structure of partitions while preserving key properties, including planarity, connectivity, and undirectedness. The black vertices, representing internal points within each zone, reinforce the completeness of the zones and contribute to the overall graph stability.

A group of circles with dots and lines

Description automatically generated

Figure 4: Graph Perturbation After One Iteration

Figure 4 presents the result of a single iteration of perturbation applied to the graph depicted in Figure 3. In this iteration, a boundary vertex initially located within Zone B, represented in green, is reassigned to Zone C. This vertex's movement reflects the perturbation process, where boundary vertices shift between adjacent zones to explore new partition configurations.

Following the perturbation, the vertices in Zone B that were previously connected to the reassigned vertex are now designated as boundary vertices. These vertices, which were internal prior to the perturbation, are highlighted in green to indicate their new status as boundary vertices of Zone B. This visual distinction emphasizes the dynamic nature of the boundary during perturbation, showcasing how the reassignment of a single vertex can influence the boundary structure of neighboring zones.

The figure effectively illustrates the adaptability of the graph under perturbation, while maintaining its essential properties—connectedness, planarity, and undirectedness. The visualization highlights the interconnectedness between zones and the crucial role boundary vertices play in ensuring the graph's integrity during iterative modifications.

## **Complexity Analysis of Algorithms**

The complexity analysis evaluates the time requirements of the algorithms used in comparing the similarity between two territorial partitions. Each algorithm is analyzed step by step, using Big notation to describe its computational cost. This section integrates these analyses to derive the overall complexity of the partition comparison process.

The **Partition Comparison and Similarity Evaluation** algorithm begins by initializing the comparing graph, , as a reference to the root graph, . This initialization involves a simple assignment operation and thus has a constant time complexity of . The subsequent step applies perturbation to, for iterations using the **Perturb Partition** algorithm. Perturbation modifies the graph by reassigning vertices between zones while preserving properties such as connectedness and planarity. The complexity of perturbation per iteration depends on updating boundary vertices, which involves examining each vertex and its neighbors. The overall complexity of perturbation for iterations is , where is the number of vertices (elementary units), and is the number of edges in the graph.

Following perturbation, the algorithm iterates through each zone in both and to retrieve the identifiers of elementary units and update the weights for each zone. Retrieving identifiers involves scanning all elementary units, which has a time complexity of . Similarly, updating weights, which aggregates the characteristics of elementary units, also has a complexity of to this stage.

The algorithm then constructs the contingency table to represent the relationships between zones in and . The contingency table, which has a size proportional to the square of the number of zones , is constructed by comparing zones pairwise, leading to a time complexity of . Following this, the algorithm computes the traditional pair counting components and their weighted counterparts using the contingency table. Both operations involve accessing and processing all entries in the table, which results in a complexity of . The computation of the similarity index, a straightforward calculation based on these components, has a constant complexity of .

The **Update Boundary Vertices** algorithm is a key subroutine in perturbation. It starts by clearing the existing boundary vertices for a zone, which is a constant-time operation. The algorithm then iterates through all vertices in the zone, and for each vertex, it inspects its neighbors to determine whether they belong to different zones. If a neighboring vertex is in a different zone, the current vertex is designated as a boundary vertex, and the neighboring zone is added to its list of neighboring zones. Assuming each zone has vertices on average, and each vertex has neighbors, the complexity per zone is . Considering all zones, the overall complexity of updating boundary vertices is , where is the average degree of a vertex.

The **Perturb Partition** algorithm builds on the boundary vertex updates and performs perturbation for iterations. Each iteration involves selecting a zone, identifying boundary vertices, and reassigning a boundary vertex to a neighboring zone. These steps are dominated by the cost of updating boundary vertices, which is per iteration. Consequently, for iterations, the overall complexity of the perturbation algorithm is .

Combining these analyses, the overall complexity of the **Partition Comparison and Similarity Evaluation** algorithm can be expressed as the sum of the perturbation complexity, weight updates, and similarity computations. Perturbation contributes , weight updates contribute , and the contingency table construction and pair counting components contribute . Therefore, the total complexity of comparing two partitions is approximately . This result highlights that complexity is primarily influenced by the perturbation steps for large graphs and by the number of zones for dense partitions comparisons.

# 

# Numerical Experiments and Behavior of the Indices

This section presents the numerical experiments conducted to evaluate the behavior and performance of the similarity indices introduced in this thesis. The goal of these experiments is to analyze how the indices respond to variations in territorial partitions and to validate their effectiveness in measuring similarity under different perturbations scenarios. By applying the developed algorithms to generated data sets, we aim to provide empirical evidence supporting the theoretical foundations of weighted pair counting indices, as well as their applicability to real-world.

Through systematic perturbation of partitions, the experiments assess the sensitivity of indices such as the Rand Index, Fowlkes-Mallows Index, and Jaccard Index to incremental changes in partition structures. This section also explores how the introduction of weight influences the indices, offering insights into their ability to capture nuanced differences between partitions.

The experiments are structured to ensure reproducibility and scalability, reflecting realistic districting scenarios. The following subsections details the process of data set generation, preprocessing, and the specific methodologies applied to simulate and measure the behavior of the indices across various partitions configurations.

## **Data Set Generation and Preprocessing**

The data set used in this thesis is derived from real-world geographical data, specifically representing the continental region of Portugal. This region is divided into 18 districts, which serve as zones, and further subdivided into 278 municipalities, corresponding to the elementary units (vertices) of the graph. Figure 5 illustrates the labeled map of Portugal, highlighting the division into districts and municipalities.

Each municipality is modeled as a vertex within the graph, and adjacency relationships between municipalities are represented as edges connecting the vertices. The resulting graph, denoted as , consists of 278 vertices and 541 edges, ensuring the graph is undirected, connected, and planar, reflecting the real-world contiguity of municipalities.

The characteristic selected for experimentation is the population of each municipality, represented by the function , which returns the population of elementary unit . This characteristic plays a crucial role in the weighting process of zones and influences the computation of weighted pair counting indices during partition comparison.

The data set generation begins by defining the zones, which correspond to the 18 districts of Portugal. A list of zones, passed as input to the system, initiates the partitioning process. Subsequently, elementary units (vertices) are created based on a list of municipalities, also provided in the input. This list contains the identifier of each municipality, the population value, and the zone to which the municipality belongs.

To construct the contiguity graph, a map structure is utilized, where each key represents the identifier of a municipality, and the corresponding value is a list of identifiers representing adjacent municipalities. This adjacency list forms the backbone of the graph, ensuring that each vertex is connected to its neighboring vertices, accurately modeling the geographical boundaries between municipalities.

The construction process systematically iterates over the input data, adding vertices to their respective zones while simultaneously connecting them based on adjacency relationships. Because the input reflects the real-world geographical structure of Portugal, the resulting graph inherently preserves the undirected, connected, and planar properties of the original map.

Ensuring the preservation of fundamental graph properties throughout the data generation and perturbation processes is critical for maintaining the integrity of the partitioning analysis. Since the initial map of Portugal is inherently undirected, connected, and planar, the generated graph retains these attributes. Perturbations applied during the experimentation phase, as described in the perturbation section, are carefully designed to maintain these essential properties, preventing the creation of disconnected or overlapping partitions.

Once the graph is constructed, preprocessing steps are conducted to validate the data and ensure consistency. This includes verifying the completeness of adjacency relationships, resolving any discrepancies in the input data, and normalizing population characteristics where necessary. Preprocessing ensures that the graph is free of isolated vertices and that all zones contain contiguous municipalities, aligning with the districting requirements of the experiment.

The resulting data set serves as a robust foundation for partition comparison experiments, allowing for realistic simulations and evaluations of territorial partitioning across Portugal. By leveraging real-world geographical data, the thesis ensures that the experiments not only reflect theoretical constructs but also address practical districting challenges faced in political and administrative decision-making.

## **Results**

# 

# Conclusion

## **Conclusions**

## **System Limitation and Future Work**

# Bibliography

|  |  |
| --- | --- |
| [1] | J. P. R. G. P. Da Silva, "Master’s Thesis: Integrality, Contiguity, and Absence of Holes in forming Political Districting maps," Lisbon, 2023. |
| [2] | F. T. Pereira, V. Mousseau and J. R. Figueira, "Multiple Criteria Districting Problems, Models, Algorithms, and Applications," ResearchGate, Coimbra, Portugal, 2014. |
| [3] | F. Liberatore, M. Camacho-Collados and B. Vitoriano, "Police Districting Problem," in *Optimal Districting and Territory Design (International Series in Operations Research & Management Science)*, vol. 284, Springer; 1st ed. 2020 edition, 2020, p. 9–29. |
| [4] | F. T. Pereira, J. R. Figueira, V. Mousseau and B. Roy, "Comparing two territory partitions in districting problems," *Socio-Economic Planning Sciences,* vol. 43, no. 1, pp. 72-88, 2009. |
| [5] | W. J. Matthijs and V. H. Hanneke, "Understanding the Adjusted Rand Index and Other Partition Comparison Indices Based on Counting Object Pairs," *Springer,* vol. 39, no. 3, pp. 487-509, 2022. |
| [6] | S. R. Department, "Big five- statistics and facts," Statista, 2024. [Online]. Available: https://www.statista.com/topics/5909/-big-five/#editorsPicks. |
| [7] | soccerphile, "soccerphile.com," soccerphile, 2018. [Online]. Available: https://www.soccerphile.com/soccerphile/news/golden-boot.html. |
| [8] | H. Song, P. Jeff, T. John and D. J. William, "Learning both Weights and Connections for Efficient Neural Networks," in *NIPS 2015*, 2015. |
| [9] | R. K. M. Fernando and T. P. Chris, "Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks," in *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, 2022. |
| [10] | H. Y and F. A, "Exploration of Common Clustering Methods and the Behavior of Certain Agreement Indices," *Ball State Undergraduate Mathematics Exchange,* vol. 12, pp. 35-50, 2018. |
| [11] | C. Marjan and F. Anuska, "Comparing Two Partitions of Non-Equal Sets of Units," *Advances in Methodology and Statistics,* vol. 15, no. 1, pp. 1-21, 2018. |
| [12] | H. Michaela, S. Douglas and B. J. Michael, "A Note on Using the Adjusted Rand Index for Link Prediction in Networks," *Social Networks,* vol. 42, pp. 72-79, 2015. |
| [13] | H. Eyke and R. Maria, "A Fuzzy Variant of the Rand Index for Comparing Clustering Structures," in *International Fuzzy Systems Association World Congress and European Society of Fuzzy Logic and Technology Conference, IFSA-EUSFLAT*, Lisbon, 2009. |
| [14] | W. J. Matthijs and v. d. H. Hanneke, "Understanding the Rand Index," in *Advanced Studies in Classification and Data Science*, Springer, Singapore, 2020, p. 301–313. |
| [15] | A. Sonia, d. Antonio, C. Iorio and S. Roberta, "Adjusted Concordance Index: an Extensionl of the Adjusted Rand Index to Fuzzy Partitions," *JOURNAL OF CLASSIFICATION,* vol. 38, no. 1, p. 112–128, 2020. |
| [16] | A. N. Ahmed and M. Niewiadomska-Bugaj, "Correcting Jaccard and other similarity indices for chance agreement in cluster analysis," *Advances in Data Analysis and Classification,* vol. 5, no. 3, p. 179–200, 2011. |
| [17] | Y. Farah and W. S. Tjong, "Optimization in Time and Score using IID Algorithm for K-Modes Clustering," *Building of Informatics, Technology and Science (BITS),* vol. 4, no. 4, pp. 1705-1713, 2023. |
| [18] | M. Farzan, Y. S. Hooman, F. Bita, B. Malihe, K. S. Zohreh and B. Sajjad, "Tutorial on Evaluation Indices of Statistical Performance: A Review Study," *Journal of Community Health Research,* vol. 12, no. 4, pp. 25-29, 2023. |
| [19] | S. S. Shankara, K. Maithilib, K. Madhavi and S. B. Yashwan, "Evaluating Clustering Algorithms: An Analysis using the EDAS Method," in *15th International Conference on Materials Processing and Characterization (ICMPC 2023)*, 2023. |
| [20] | B. A. N and T. A. Yu, "Generalization of Jaccard Index for Interval Data Analysis," *Measurement Techniques,* vol. 65, no. 12, p. 882–890, 2023. |
| [21] | O. Daniel and E. M. Elizabeth, "Jaccard Index in Ensemble Image Segmentation: An Approach," in *CIIS 2022: 2022 The 5th International Conference on Computational Intelligence and Intelligent Systems*, Quzhou, China, 2022. |
| [22] | L. d. C. Fontoura, "On similarity," *Statistical Mechanics and its Applications,* vol. 599, no. C, 2022. |
| [23] | G. Martijn, T. Alexey and P. Liudmila, "Systematic Analysis of Cluster Similarity Indices: How to Validate Validation Measures," *Proceedings of the 38th International Conference on Machine Learning, PMLR,* vol. 139, pp. 3799-3808, 2021. |
| [24] | S. Kosub, "A note on the triangle inequality for the Jaccard distance," *Pattern Recognition Letters,* vol. 120, no. C, pp. 36-38, 2019. |
| [25] | S. K. Saha, "A Relevance Feedback Approach for Social Network Clustering in the Context of Triangle Inequality Violations," Brandenburgische Tecnische Universitat Cottbus, Senftenberg, 2023. |
| [26] | S. Merendino and M. Emre Celebi, "A Simulated Annealing Clustering Algorithm Based on Center Perturbation Using Gaussian Mutation," in *Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference*, St. Pete Beach, Florida, USA, 2013. |
| [27] | Y. Wang, Y. Shi and Y. Liu, "Research on improved genetic simulated annealing algorithm for multi-UAV cooperative task allocation," in *2022 8th International Symposium on Sensors, Mechatronics and Automation System*, Online, 2022. |
| [28] | Y. Lei, J. C. Bezdek, S. Romano, N. X. Vinh, J. Chan and B. James, "Ground truth bias in external cluster validity indices," *Pattern Recognition,* vol. 65, no. C, pp. 58-70, 2017. |

Appendix

|  |  |
| --- | --- |
| **Index (Abbreviation)** | **Expression** |
| Rand Index (*R*I) |  |
| Adjusted Rand (*AR*) |  |
| Jaccard (*Jaccard*) |  |
| Jaccard Distance (*JD*) |  |
| Wallace1 (*W*) |  |
| Wallace2 |  |
| Dice |  |
| Correlation Coefficient (*CC*) |  |
| Correlation Distance (*CD*) |  |
| Sokal&Sneath-I (*S&S*) |  |
| Minkowski |  |
| Hubert (*H*) |  |
| Fowlkes&Mallow (*FM*) |  |
| Sokal&Sneath-II |  |
| Normalized Mirkin |  |
| Kulczynski |  |
| McConnaughey |  |
| Yule |  |
| Baulieu-I |  |
| Russell&Rao |  |
| Fager&McGowan |  |
| Peirce |  |
| Baulieu-II |  |
| Sokal&Sneath-III |  |
| Gower&Legendre |  |
| Rogers&Tanimo |  |
| Goodman&Kruskal |  |

Table 4: A selection of pair counting indices. Most of these indices are taken from [26]

A map of the state of portugal

Description automatically generated

Figure 5: Portugal Continental Map