A generic approach to districting with diameter or center-based objectives

Alex Gliesch
Federal University of Rio Grande do Sul
Brazil
alex.gliesch@inf.ufrgs.br

Marcus Ritt
Federal University of Rio Grande do Sul
Brazil
marcus.ritt@inf.ufrgs.br

ABSTRACT

Districting is the problem of grouping basic geographic units into clusters called districts. Districts are typically required to be geometrically compact, contiguous, and evenly balanced with respect to attributes of the basic units. Though most applications share these core criteria, domain-specific constraints and objective functions are common. This leads to a fragmented literature with different approaches for similar domains and hinders comparisons between these approaches. In this paper we study a unified heuristic approach that can handle three of the most common objective functions concerning compactness: diameter, p-center and p-median, as well as a variable number of balancing attributes. We propose a multistart method which iteratively constructs greedy randomized solutions followed by an improvement phase which alternates between optimizing compactness and satisfying balancing constraints through a series of tabu searches. Experiments show that the proposed method is competitive when compared to approaches in the literature which are specific to each objective, improving known upper bounds in some cases.

CCS CONCEPTS

• **Computing methodologies** → **Search methodologies**; • **Applied computing** → **Operations research**;

KEYWORDS

Districting, Territory Design, Hybrid heuristic, *p*-median, *p*-center, diameter.

ACM Reference Format:
Alex Gliesch and Marcus Ritt. 2019. A generic approach to districting with diameter or center-based objectives. In *Genetic and Evolutionary Computation Conference (GECCO '19), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3321707.3321874

1 INTRODUCTION

Districting is the problem of grouping basic geographical units into clusters, also called districts. Basic units usually correspond to contiguous real-world regions of similar size or category, such as city blocks, streets or parcels of land. Districting has a broad range of applications, including electoral districting [3, 13, 18, 21, 24, 26], design of commercial territories [12, 22, 31, 34], agrarian lots [2, 14], service districting such as police [6, 8], energy [42], salt spreading [25], waste collection [17] or health care districts [40]. Ref. [20] provides a comprehensive overview of the different applications as well as solution methods to districting problems.

Almost all districting problems share three main requirements:

(1) *Connectivity*: districts must be connected regions in space. Connected districts generally reduce travel times of agents and, in the case of political districts, are often legally required. Connectivity is almost always treated as a hard constraint.
(2) *Balancing*: districts must be evenly balanced with respect to attributes of the basic units, such as population, product demand or expected service time. Some applications have multiple balancing constraints.
(3) *Compactness*: districts should have a geometrically convex and compact shape, not being too long or narrow. Compactness is generally seen as a subjective concept. Measures differ widely from domain to domain (see [5] for an overview), and are often specially designed for the particular application. Compact districts tend to have shorter inner-district travel times and, in the case of political districting, help prevent gerrymandering.

In addition to these core criteria, domain-specific requirements are common. Examples are maximizing the similarity to past plans [3, 32], forbidding enclaves [13, 21], satisfying conflict relations between the basic units [32], or maximizing the number of districts subject to a minimum district size [9].

Problems have been modeled in a number of different ways in the literature. Some works use an objective function which is a convex combination of multiple criteria, possibly including balancing and several different measures of compactness [3]. Others consider a multi-objective approach [1, 27, 35]. The most common approach, however, is to optimize compactness as the objective function and treat the remaining criteria as constraints, where they are limited by a maximum allowed value. Balancing, for example, is usually handled by limiting the maximum deviation of a district's total or average attribute value to a target value.

The wide range of compactness measures, the domain-specific requirements and several different modeling techniques make each problem in the literature more or less "unique". As a consequence, solution methods are often tailored to each model and there is a lack of standard instances sets or cross-domain comparisons between different approaches.

GECCO '19, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6111-8/19/07…$15.00
https://doi.org/10.1145/3321707.3321874

In this paper we take a first step towards the unified solutions of these problem variants. We propose a generic single-objective heuristic solver that obtains connected and balanced districts and that can handle the three most common objective functions found in the literature: the center-based approaches p-median and p-center, where a center unit is assigned to each territory, and the diameter, which is independent of centers. For all three objectives the desired number p of districts is given. The p-median objective minimizes the sum of the distances between the units in a district and their center, summed over all districts. The p-center minimizes the maximum distance between any unit and its center. The diameter minimizes the maximum distance between two same-district units. For the p-center and p-median approaches, centers that minimize the respective objective function can be found in linear time. The method we propose in this paper can be applied to either objective function and multiple balancing constraints with little modification. It iteratively constructs solutions by a greedy randomized algorithm, and then improves them by an algorithm which alternates between a tabu search aimed at improving compactness and a series of constrained tabu searches aimed at balancing the solution.

The rest of this paper is structured as follows. Section 2 gives an overview of related work in districting concerning the three objective functions. Section 3 introduces the problem formally. Section 4 and subsections therein present the proposed algorithm in detail. We discuss some relevant implementation details in Section 5. In Section 6 we calibrate the parameters of our algorithm and compare its effectiveness with existing approaches in the literature. We conclude in Section 7.

2 RELATED WORK

Table 1 summarizes the main studies which consider either p-median, p-center or diameter in the objective function. For each paper we present the application domain, the objective function and main constraints, including the number of balancing criteria and other domain-specific constraints, the solution method used and the sizes of the instances considered, where n denotes the number of units and p the number of desired districts. All studies except [42] and [10] treat connectivity as a hard constraint and, except [13], use compactness as (part of the) objective function.

Ref. [13] propose an exact method based on set partitioning where a first step generates a set of feasible districts, and a second step selects them so as to minimize imbalance. It is one of the only works in the literature which considers balancing as an objective, and treats compactness by setting an upper limit on the diameter/area ratio of each district. A similar approach is used by [24] who propose a column generation framework to solve a model with the p-median objective; however, the corresponding pricing problem is still too difficult to be solved optimally in feasible time, so their method is not exact.

Ref. [34] propose exact solutions by integer programming (IP) models for both p-center and p-median objectives with two balancing constraints. The models do not consider district connectivity directly, and connectivity cuts are added iteratively as optimal unconnected solutions are encountered by the solver. A similar technique is also used by [32, 35] to solve problems with composite objectives.

Refs. [12, 38, 42] propose heuristic solutions based on the location-allocation originally proposed by [18], which iteratively selects district centers with a heuristic and then allocates the remaining units optimally, given the fixed centers. Concerning metaheuristic methods, local search-based methods such as greedy randomized adaptive search procedures (GRASP) [28, 30, 31, 33, 36], simulated annealing [1] and tabu search [15] are the most common, and typically use a neighborhood structure which changes the assignment of a basic unit from its current district to an adjacent district. Refs. [1, 35–37] propose heuristic solutions to bi-objective models which optimize one balancing attribute and the p-median value as objective, while treating additional requirements as constraints.

With respect to instance sizes, current heuristic approaches can usually handle instances of up to 10000 basic units, whereas exact approaches are limited to about 200 basic units.

3 DEFINITIONS

In the following, we use the set notation $[n] = \{1, \dots, n\}$. We are given an undirected planar graph $G = (V, E)$ with $|V| = n$ nodes, the desired number of districts $p \leq n$, and a balancing threshold $\tau \in [0, 1]$. We use the terms node and basic unit interchangeably. Each unit $v \in V$ is associated with a activity values $w_v^i \in \mathbb{R}, i \in [a]$. The distance between nodes u and v is given by $d_{uv} \in \mathbb{R}$.

A districting plan (or solution) S is a partition $S_1 \dot\cup \cdots \dot\cup S_p \subseteq V$ of the nodes into p districts. We say that solution S is *complete* if $\dot\bigcup_{k\in[p]} S_k = V$, otherwise it is *incomplete*. We also use the notation $S(v) = k$ to denote that node v is *assigned* to district k, i.e. $v \in S_k$. If S is incomplete, then $S(v)$ is undefined for some v, in which case we call v *unassigned*. We define the *boundary* $\partial S_k = \{v \mid v \notin S_k, \exists u \in S_k : \{u, v\} \in E\}$ of district k as the set of nodes not in S_k which are directly adjacent to S_k. We say that two districts k and l are *neighbors* if $S_k \cap \partial S_l \neq \emptyset$ (or alternatively $\partial S_k \cap S_l \neq \emptyset$, since the graph is undirected).

The target value $\mu_i = \sum_{v\in V} w_v^i/p$ of each district with respect to activity $i \in [a]$ is given by the total value of activity i in V, averaged over the p districts. Let $b_i(S_k) = |(\sum_{u\in S_k} w_u^i - \mu_i)/\mu_i|$ denote the absolute relative deviation of activity i in district k to the target μ_i. We say that district k is *balanced* with respect to activity i if $b_i(S_k) \leq \tau$, and that k simply is balanced if it is balanced with respect to all $i \in [a]$. A solution is balanced if all its districts are balanced. The *imbalance* of district k denotes its total balance violation and is given by $b(S_k) = \sum_{i\in[a]} \max\{0, b_i(S_k) - \tau\}$. Similarly, the imbalance of solution S is $B(S) = \sum_{k\in[p]} b(S_k)$.

We denote the *compactness* of a solution S by $C(S)$. Depending on which objective function is being considered, $C(S)$ will represent the diameter, the p-center value or the p-median value of a solution S. The diameter $diam(S) = \max\{d_{uv} \mid u, v \in V, S(u) = S(v)\}$ of solution S is the maximum distance between two nodes belonging to the same district. The p-center value $pcen(S) = \max_{k\in[p]} \max_{u\in S_k} d_{uc_k}$ is the maximum distance of a node $u \in S_k$ to its district center c_k, for any district k. We assume that the district centers are placed optimally, i.e. $c_k = \arg\min_{v\in S_k}\{\max_{u\in S_k} d_{uv}\}$. The p-median objective value $pmed(S) = \sum_{k\in[p]} \sum_{u\in S_k} d_{uc_k}$ is the sum of distances of each node to its district center. As before, we assume the district centers are placed optimally, i.e. $c_k = \arg\min_{v\in S_k}\{\sum_{u\in S_k} d_{uv}\}$.

Table 1: Works in the literature that consider districting with center-based or diameter-based objective functions

Reference	Year	Domain	Objective(s)	Constraint(s)	Solution method	Instances
Garfinkel and Nemhauser [13]	1970	political	1 bal. attr.	diameter/area	Set partitioning	$n \in [26, 55]$, $p \in [5, 7]$
Fleischmann and Paraschis [12]	1988	commercial	p-median	1 bal. attr.	Location-allocation	$n = 1402$, $p = 168$
Mehrotra et al. [24]	1998	political	p-median	1 bal. attr.	Branch-and-cut (heu.)	$n = 46$, $p = 6$
Bergey et al. [1]	2003	power distr.	(p-median, 1 bal. attr.)		Simulated annealing	$n = 30$, $p = 5$
Segura-Ramiro et al. [38]	2007	commercial	p-median	2 bal. attr.	Location-allocation	$n \in \{500, 1000\}$
Ríos-Mercado and Fernández [31]	2009	commercial	p-center	3 bal. attr.	GRASP	$n = 500$, $p = 10$
Ríos-Mercado and Salazar-Acosta [33]	2011	commercial	diameter	2 bal. attr., max. distr. TSP	GRASP + strategic osc.	$n = 1000$, $p = 40$
Salazar-Aguilar et al. [34]	2011	commercial	p-median ∨ p-center	2 bal. attr.	IP + cuts	$n \in [60, 200]$, $p \in [4, 11]$
Salazar-Aguilar et al. [35]	2011	commercial	(p-median, 1 bal. attr.)	2 bal. attr.	IP + cuts	$n \in [60, 150]$, $p \in [4, 6]$
Salazar-Aguilar et al. [37]	2012	commercial	(p-median, 1 bal. attr.)	1 bal. attr.	Scatter search	$n \in \{500, 1000\}$, $p \in \{20, 50\}$
Ríos-Mercado and López-Pérez [32]	2012	commercial	p-median + sim. to ex. plan	3 bal. attr. conflicts dif. to cur. plan	IP + cuts	$n \in \{5000, 10000\}$, $p = 50$
Chou et al. [7]	2012	political	diameter	1 bal. attr.	Genetic algorithm	$n = 66$, $p = 10$
Salazar-Aguilar et al. [36]	2013	commercial	(p-median, 1 bal. attr.)	1 bal. attr.	GRASP	$n \in [500, 1000]$, $p \in [20, 50]$
Elizondo-Amaya et al. [10]	2014	commercial	p-center	2 bal. attr. no conn.	Dual bounding	$n \in [60, 2000]$, $p \in [4, 20]$
Ríos-Mercado and Escalante [30]	2016	commercial	diameter	3 bal. attr.	GRASP + path relinking	$n = 500$, $p = 10$
Yanik et al. [42]	2016	energy supply system	p-median	1 bal. attr. no conn.	Location-allocation	$n = 933$, $p \in [7, 100]$
Ríos-Mercado [28]	2016	commercial	p-center	3 bal. attr.	GRASP	$n \in \{1000, 2000\}$, $p = 20$
Gliesch et al. [15]	2018	commercial	diameter	3 bal. att.	Tabu search	$n \in [500, 10000]$, $p \in [10, 160]$

We call a solution S *feasible* if it is balanced, complete, and the subgraph of G induced by each district S_k is connected. Our goal is to find a feasible solution which minimizes $C(S)$. We use the pair notation $BC(S) = (B(S), C(S))$ to indicate a lexicographical comparison of solutions with respect to non-increasing balance which breaks ties by non-increasing compactness, and similarly $CB(S) = (C(S), B(S))$ for an order that gives preference to more compact solutions, breaking ties by balance.

Finally, we define two modification operators for complete solutions. A *shift* $v \rightarrow k$ applied to a solution S changes the assignment of node $v \in V \setminus S_k$ to district $k \in [p] \setminus \{S(v)\}$. Similarly, a *swap* $v \leftrightarrow u$ exchanges the currently assigned districts of nodes v and u, $S(u) \neq S(v)$. The notation $S[m]$ is used to refer to the solution obtained by applying operator m to solution S.

Algorithm 1 Main algorithm.

```
1:  R ← ∅
2:  repeat
3:      S ← selectInitialSolution()
4:      X = {S}
5:      for i ∈ [A_max] do
6:          S' ← optimizeCompactness(S)
7:          S' ← optimizeBalance(S', S)
8:          if B(S') > 0 or S' ∈ X then
9:              break
10:         S ← S', X ← X ∪ {S}
11:     R ← arg min{BC(R), BC(S)}
12: until time limit reached
13: return R
```

4 PROPOSED ALGORITHM

The multistart heuristic we propose here is outlined in Algorithm 1. At each iteration we generate a number of initial solutions by a randomized greedy constructive heuristic, and select a high-quality one using a simple filtering mechanism (line 3). Section 4.1 gives further details about this step.

Next, we iteratively alternate between two procedures aimed at improving compactness and balance (lines 6–7), in this order. At each alternation, we store the current solution in a hash table X, and move on to the next multistart iteration if cycling is detected (lines 8-9). Since the total number of intermediate solutions is expected to be small, this is not performance-critical. We also stop when the solution is still infeasible after an attempt to balance it, since it is unlikely that it will become balanced in a subsequent iteration, or after a maximum number A_{max} of alternations. We use parameter A_{max} to avoid alternating indefinitely between low and high compactness values without a global improvement. In practice, this happens rarely. In our implementation, we have set $A_{max} = 100$.

Both alternating procedures are based on tabu search. Tabu search is a non-monotone local search proposed by [16]. It iteratively executes the best neighboring operator on the current solution, with respect to some objective. If multiple candidates exist, we select one uniformly at random. To avoid cycling, some operators are declared tabu after a neighboring move, and cannot be selected again for t iterations, where the tenure t is a parameter. Specifically, after executing a shift $u \rightarrow k$ we mark operators $u \rightarrow i, i \in [p]$ and $u \leftrightarrow v, v \in V$ incident to u as tabu; similarly, after a swap $u \leftrightarrow v$ we mark all operators incident to u or v as tabu. We discard all moves which violate district connectivity. The search terminates after $I_{\max}$ iterations without improvement, where $I_{\max}$ is a parameter, and

returns the best intermediate solution w.r.t. *BC* if we are optimizing for balance, or *CB* if we are optimizing for compactness.

We improve compactness with a standard tabu search on a specific neighborhood for each objective. These neighborhoods are explained in further detail in Sections 4.3, 4.4, and 4.5. To improve balance, we execute a series of tabu searches on neighborhoods bounded by a maximum increase to compactness. This is done with the intent to maintain the progress in compactness made so far. Given an upper bound $C(S)$, where S is the previous solution before optimizing compactness (line 7), we apply a binary search to find the smallest $c^* \in [C(S'), C(S)]$ for which a tabu search aimed at reducing imbalance and bounded by $C \leq c^*$ obtains a feasible solution. Section 4.2 explains the details of this procedure. Both optimization procedures always maintain connectivity.

Note, that the alternated search strategy is agnostic of the choice of optimization algorithms for balancing or improving compactness. We use tabu search, as it has been effective in solving related problems in the past [3, 15, 29], but other methods might also be used without loss of generality.

4.1 Initial solutions

We use the constructive heuristic of [15] to generate new solutions. Although this heuristic was originally developed for the diameter objective, we found in preliminary tests that it also produced satisfactory results for the other two objectives without further adaptations, compared to simpler approaches. It consists of two phases. First, p seed nodes, one for each district, are obtained by a randomized greedy heuristic to the p-dispersion problem [11], which aims at maximizing the minimum distance between two units. Next, starting from a basic solution where each seed is assigned to a district, it iteratively selects a district k of minimum total weight $\sum_{i\in[a]} \sum_{u\in S_k} w_u^i$ and assigns to it an unassigned node $v \in \partial S_k$ which minimizes $C(S[v{\to}k])$. If there are multiple candidate units, one is selected uniformly at random. For more details on the algorithm as well as an experimental discussion, we refer the reader to [15].

4.1.1 Filtering. In early experiments we have found that the quality of the solution obtained after the alternating procedure is highly dependent on the compactness of the initial solutions. Since the alternating phase is significantly more time-consuming than the construction phase, we filter low quality initial solutions in order to invest time in more promising ones. To this end, at each multistart iteration we construct p randomized greedy solutions and add them to a pool P of possible initial solutions. We then select the solution $S \in P$ with minimum $CB(S)$ to be improved next, and remove S from P. The pool P is carried on to the next iteration. To limit memory usage, we limit $|P|$ to $2p$ and discard the $\max\{0, |P| - 2p\}$ worst solutions w.r.t. *CB* at each iteration. Here we are greedy w.r.t. *CB* as opposed to *BC*, since empirically the initial imbalance of a solution is not a good predictor of the difficulty to balance it in the alternating phase.

4.2 Optimizing balance

We balance solutions by executing a series of tabu searches on restricted neighborhoods, which are defined by setting an upper limit to the compactness of a solution. Specifically, given a solution

Algorithm 2 Improving balance

```
1:  procedure improveBalance(S, S_u)
2:      U ← C(S_u)
3:      if B(S_u) > 0 then
4:          U ← 2U
5:      L ← C(S), R ← S
6:      while U > L ∧ B(S) ≠ 0 do
7:          S' ← TS(S, (U + L)/2)
8:          if B(S') = 0 then
9:              R ← S', U ← C(R) − d_min
10:         else
11:             S ← S', L ← (U + L)/2 + d_min
12:     return R
```

S to be balanced and an upper limit $c \geq C(S)$, the respective tabu search $TS_b(S, c)$ considers neighborhoods $N^b_{shift}(S, c) = \{u{\to}k \mid k \in [p], u \in \partial S_k, C(S[u{\to}k]) \leq c\}$ and $N^b_{swap}(S, c) = \{u{\leftrightarrow}v \mid S(u) \neq S(v), u \in \partial S_{S(v)}, v \in \partial S_{S(u)}, C(S[u{\leftrightarrow}v]) \leq c\}$, in this order. At each iteration, we first look for non-tabu shifts $m \in N^b_{shift}$ which minimize $BC(S[m])$, and that are improving, i.e. $B(S[m]) < B(S)$. If such a move is found, we apply it immediately and move on to the next iteration; otherwise, we search for swaps in N^b_{swap} under the same criteria. We then apply the best move found (among all shifts and swaps) w.r.t. *BC*. To consider first shifts, then swaps in order has been generally effective for grouping problems [4, 19]. Since the size of region boundaries in planar domains tends to scale closely to $\sqrt{n}$, generally $|N^b_{shift}| \approx \sqrt{n}$ while $|N^b_{swap}| \approx n$; in practice, visiting all N^b_{swap} at each iteration in a search for the best possible move is not worth the cost.

The improvesBalance procedure is given a (likely imbalanced) initial solution S and the solution S_u incident to the current upper bound (i.e. the best solution visited in this multistart iteration), such that $C(S) \leq C(S_u)$. Let $U = C(S_u)$, or, if this is the first call to improvesBalance (i.e. S_u may not be balanced), $U = 2C(S)$. We apply binary search to find the smallest $c^* \in [C(S), U)$ such that $TS_b(S, c^*)$ is balanced. This is motivated by the fact that $B(TS_b(S, c))$ is expected to be monotonically decreasing as c grows: given some c_1 and c_2 such that $c_1 < c_2$, the neighborhood explored by search $TS_b(S, c_2)$ is a superset of the one explored by $TS_b(S, c_1)$, and thus $TS_b(S, c_2)$ typically has more available improving movements. Because we already have a feasible solution of value U, we also expect that $TS_b(S, U)$ will be balanced. Of course, since TS_b is a heuristic algorithm, this may not always be the case.

Algorithm 2 outlines the complete balancing method. At each step of the binary search with upper and lower limits U and L, we execute a constrained tabu search with $c = (U + L)/2$ (line 7). If the resulting solution is balanced, we update the return solution R as well as the upper limit $U = C(R) - d_{min}$ (line 9), where $d_{min} = \min\{d_{uv} \mid u, v \in V\}$ is the smallest distance between any two units; otherwise, we update the current solution S and the lower limit $L = (U + L)/2 + d_{min}$ (line 11). We used step size d_{min} since we found it empirically to be a good compromise between precision and the total number of calls to the tabu search TS.

4.3 Optimizing p-median

We optimize the p-median objective with a standard tabu search on neighborhood $N^{pm} = \{u \to k \mid k \in [p], u \in \partial S_k\}$. This neighborhood is similar to N^b_{shift} in that it considers all possible shifts of nodes to adjacent districts. We have also considered a neighborhood which uses swap moves, but we have found it to be not effective.

4.4 Optimizing p-center

We optimize the p-center objective with a standard tabu search which considers two different neighborhoods in an order, in the same way as the balancing tabu search TS_b. To explain them, we first define the set of *critical units* $K_u(S) = \{u \in V \mid d_{uc_{S(u)}} = C(S)\}$ of solution S as the set of units whose distance to their district center is equal to $C(S)$, and the set of *critical districts* $K_d(S) = \{S(u) \mid u \in K_u(S)\}$ of S as the set of districts that contain a critical node.

The first neighborhood $N^{pc}_1 = \{u \to k \mid k \in [p], u \in K_u(S) \cap \partial S_k\}$ considered consists of shifting critical units out of their critical districts, in an attempt to reduce the local compactness of their district and consequently the global compactness of the solution.

The second neighborhood $N^{pc}_2 = \{u \to k \mid k \in [p], u \in \{S_j \cap \partial S_k \mid j \in K_d(S)\}\}$ consists of shifting arbitrary units out of critical districts. We do so in an attempt to induce a change to the optimal district center, in such a way that it is moved closer to the critical units and therefore decreases the district's local compactness.

Similarly to TS_b, we first search for non-tabu moves $m \in N^{pc}_1$ which minimize $C(S[m])$ and, if no improving move exists for it, we then consider N^{pc}_2 under the same criteria. We consider N^{pc}_1 and N^{pc}_2 in this order since N^{pc}_1 tends to be about a factor $\sqrt{n/p}$ smaller than N^{pc}_2 and, in practice, yields the best move in about 80% of cases.

We have also conducted experiments with a third neighborhood $N^{pc}_3 = \{u \to k \mid k \in K_d(S), u \in \partial S_k\}$ in which a critical district receives a unit which will become the new optimal district center. We have found, however, that this neighborhood is seldom better than the two previous ones, yielding the best move in only 0.36% of cases, on average, and is rather time-consuming to explore fully. Therefore, we have excluded N^{pc}_3 from the final algorithm.

4.5 Optimizing diameter

We use a tabu search algorithm proposed by [15] to optimize diameters, since it solves a problem with a similar model to ours. Let $L(S) = \{u \mid \exists v \in S, S(u) = S(v), d_{uv} = D(S)\}$ be a set of *critical* units incident to the maximum diameter of S. The tabu search considers neighborhood $N^d = \{u \to i \mid i \in [p], u \in L(S) \cap \partial S_i\}$ which shifts critical units out of their current district. The main search parameters and termination criteria are the same as described at the start of Section 4. Further, [15] use an additional improvement step in their tabu search called sp-escape, in order to avoid situations where no critical node is on a boundary. We refer the reader to the original paper for more details.

5 IMPLEMENTATION

Some components of our algorithm are performance-critical and, if implemented in a naïve way, can easily dominate the running time of the algorithm. In this section, we discuss relevant data structures and speed-up techniques we used in our implementation.

5.1 Connectivity tests

Neighboring operations may break connectivity if a moved unit was an articulation node of the subgraph induced by its previous district. Testing for connectivity loss by performing a graph search on such subgraph in $O(n/p)$ time would be a bottleneck to the algorithm, since this must be done for each candidate neighbor. This can be improved by maintaining a flag on each node denoting whether it is an articulation node of its district's induced subgraph.

Articulation nodes are computed in $O(n/p)$ average time for each district with the algorithm of [41], assuming an average district size of n/p. Since these only need be recomputed after a move has been made, and only for the districts involved in the move, given a neighborhood of size k each test is done in amortized time $O(n/(kp))$. In the case of N^b_{shift}, for example, since $|N^b_{shift}| \approx \sqrt{n}$, each test takes $O(\sqrt{n}/(kp))$ time.

In the case of a swap $u \leftrightarrow v$ on solution S, we test whether u is an articulation node in $S(u)$ or v is an articulation node in $S(v)$. This can however lead us to inadvertently discard valid moves in the special case where $G \cap (S_{S(u)} \setminus \{u\})$ is not connected, but $G \cap (S_{S(u)} \setminus \{u\} \cup \{v\})$ is. In practice, however, this situation rarely happens, and the performance gained by ignoring such cases make up for the possible lost moves.

5.2 Dynamic objective updates for local search candidates

Recomputing the objective function and constraint violations for each candidate neighbor can as well become a bottleneck, especially on large neighborhoods, such as N^b_{swap}.

For balancing constraints dynamic updates are straightforward: we maintain the total value of each district with respect to each activity, update it by adding or removing values as nodes get assigned or unassigned from the district, and recompute b in constant time for each candidate.

In the next two subsections, we explain how efficient dynamic updates for the compactness measures can be achieved.

5.2.1 Dynamic p-median. We maintain, for each node $u \in V$, the compactness δ^{pm}_u of $S(u)$ if node u were the district center. Further, we keep the compactness value $C^{pm}_k = \sum_{u \in S_k} d_{uc_k}$ of each district $k \in [p]$. When considering a candidate shift $u \to k$ on solution S, the expected compactness value is given by

$$\left(\sum_{j \in [p]} C^{pm}_j\right) + \Delta_k + \Delta_{S(u)},$$

where

$$\Delta_k = \min\left\{\sum_{v \in S_k} d_{uv}, \min_{v \in S_k}(\delta^{pm}_v + d_{uv})\right\} - C^{pm}_k$$

$$\Delta_{S(u)} = \min_{v \in S_{S(u)}}(\delta^{pm}_v - d_{uv}) - C^{pm}_{S(u)}$$

are the expected changes to the compactness values of districts k and $S(u)$, respectively. Similarly, given a swap $u \leftrightarrow v$ the expected

compactness is

$$\left(\sum_{j\in[p]} C_j^{pm}\right) + \Delta_{S(v)} + \Delta_{S(u)},$$

where

$$\Delta_{S(u)} = \min\left\{\sum_{i\in S_{S(v)}} d_{iu}, \min_{i\in S_{S(v)}} (\delta_i^{pm} - d_{iv} + d_{iu})\right\}$$

(and vice-versa for $\delta^{pm}_{S(v)}$).

Iterating over the nodes of two districts for each candidate takes $O(2n/p)$ time, for both shift and swap. After a move has been made, we recompute the optimal centers as well as δ^{pm} directly in $O(n^2)$. In a neighborhood of size k, computing the p-median value of each candidate is therefore done in amortized $O(2n/p + n^2/k)$. For the case of N^b_{shift}, for example, the amortized time per candidate is then $O(2n/p + n^{3/2}) = O(n^{3/2})$.

5.2.2 Dynamic p-center. We compute dynamic p-center in a similar way to p-median. We maintain, for each node $u \in V$, two values $\delta_u^{pc,1}$ and $\delta_u^{pc,2}$ which correspond to the largest and second largest distances from u to any other node in $S(u)$ (i.e., the compactness of $S(u)$ if u were center, and the compactness of $S(u)$ if u were center and $\delta^{pc,1}$ was removed). We also keep the compactness $C_k^{pc} = \max_{u\in S_k} d_{uc_k}$ of each district $k \in [p]$. When considering a candidate shift $u\to k$ on solution S, the expected compactness value is then given by

$$\max\left\{\max_{j\in[p]\setminus\{k,S(u)\}} C_j^{pc}, \Delta_k, \Delta_{S(u)}\right\},$$

where

$$\Delta_k = \min\left\{\max_{v\in S_k} d_{uv}, \min_{v\in S_k} \max\{d_{uv}, \delta_v^{pc,1}\}\right\}$$

$$\Delta_{S(u)} = \min_{v\in S_{S(u)}} \left([d_{uv} = \delta_v^{pc,1}]\delta_v^{pc,2} + [d_{uv} \neq \delta_v^{pc,1}]\delta_v^{pc,1}\right)$$

are the expected compactness values of districts k and $S(u)$, respectively[1]. The time complexity analysis and the generalization to swaps are similar to the p-median case, and have been omitted for space reasons.

5.2.3 Dynamic diameters. We compute dynamic diameters using the same method of [15]. It maintains the convex hull of each district in $O(n/p)$ time per update and, for each candidate, obtains the new diameter in $O(\log n/p)$ time by executing the rotating calipers algorithm of [39], for a final amortized time of $O(\log n/p + n/(pk))$ in a neighborhood of size k. For N^b_{shift}, for example, this amounts to $O(\log n/p + \sqrt{n}/p)$ time.

5.3 Caching movements on TS_b

To avoid recomputing movements on TS_b, we cache the outcome of operators. Given a solution S and a maximum compactness value c, for each district $k \in [p]$ we maintain a list Λ_k of feasible operators incident to k, sorted increasing by expected BC. Further, we maintain flags λ_k that indicate whether Λ_k may contain swap moves.

[1]For a proposition P, Iverson's brackets $[P]$ evaluate to 1 if P is true, and 0 otherwise.

Initially we have $\lambda_k = 0$ and $\Lambda_k = \{u\to k \mid u \in V\} \cap N^b_{shift}(S, c)$ for all k. When computing the best neighboring move, we then proceed as follows. For each $k \in [p]$ we consider the first feasible move m of Λ_k (i.e., $S[m]$ is connected, $C(S[m]) \le c$, and $BC(S[m])$ is minimum) as a possible choice, removing infeasible moves from Λ_k in the process. If no such move improves $B(S)$, for each k such that $\lambda_k = 0$ we add to Λ_k the swap moves $\{u\leftrightarrow v \mid u, v \in V, S(u) = k\} \cap N^b_{swap}(S, c)$, sort Λ_k accordingly, and set $\lambda_k = 1$ to indicate that Λ_k now considers swaps as well as shifts. Finally, we consider once again the first feasible move in each $\Lambda_k, k \in [p]$ as a possible choice, and return the best move w.r.t. BC.

After applying a chosen move m, we reset $\lambda_j = 0$ and $\Lambda_j = \{u\to j \mid u \in V\} \cap N^b_{shift}(S, c)$ for each district j modified by m, as well as its neighbors. Because neighboring relations between districts induce a planar graph (since G is planar), the average number of districts for which Λ must be recomputed at each iteration is at most 12 (as opposed to p), since the average degree of planar graphs is at most 6. This saves us a factor p in time complexity compared to computing the entire neighborhood, but requires us to store all neighbors in memory, at the worst case. Note, however, that for the p-center and diameter objectives we only need store the first element of each Λ_k, thus using constant space, since due to the maximum-based objective a feasible move w.r.t. $C < c$ will continue to be feasible independently of subsequent moves.

6 COMPUTATIONAL EXPERIMENTS

In our computational experiments we use the domain originally introduced by [38] and subsequently studied by [10, 15, 30–37] in several variant models. It originates from a context in the distribution of goods, where basic units are city blocks and districts are independent regions of distribution. Despite the large number of variants, the corresponding models and solution methods are well-defined, and large instance sets are available. We use the models proposed by [34] for the p-median and p-center objectives, and by [30] for the diameter objective. Besides the objective function, these models have three balancing constraints concerning the attributes workload, number of customers and product demand of each city block.

6.1 Instances

We use four data sets from the literature. Data sets DS and DT comprise 20 instances of size $n = 500$ with $p = 10$ districts each and were originally introduced by [31] for a p-center-based model and are generated in such a way as to resemble real-world scenarios. Data set DS selects activity values uniformly from fixed intervals, while DT selects them from a non-uniform symmetric distribution, which tends to represent the application more closely.

Data set DL comprises four instances of each sizes $n \in \{1000, 2500, 5000, 10000\}$ and number of districts $p \in \{n/200, n/100, n/62.5\}$, for a total of 48 instances. It was introduced by [15] to assess the scalability of their heuristic algorithm to optimize the diameter. Unit weights are generated as for DL, and topologies are obtained by Delaunay triangulations.

Finally, the data set introduced by [34], which we denote by DU, contains 20 instances of each size $(n, p) \in \{(60, 4), (80, 5), (100, 6)\}$ and 10 of each $(n, p) \in \{(150, 8), (200, 11)\}$, for a total of 80 instances.

Table 2: Parameter calibration: optimization ranges and best setting found by irace.

Param.	Optimization range	p-med.	p-cen.	diam.
t	$\{0.1, 0.25, 0.5, 1, 1.5, 2, 2.5, 5\}p$	$1.5p$	$0.25p$	$5p$
$I_{\max}$	$\{50, 100, 250, 500, 1000, 2500, 5000\}$	100	1000	5000

It contains significantly smaller instances than the other data sets, as it reflects the instance sizes treatable by the exact method proposed of [34].

To be consistent with previous works, we used a balancing threshold of $\tau = 0.05$ on all instances.

6.2 Experimental setup

We performed all experiments on a PC with an 8-core AMD FX-8150 processor and 32 GB of main memory, running Ubuntu Linux 18.04. For each experiment, only one core was used. Our algorithms were implemented in C++ and compiled with GCC 7.2 with maximum optimization. The source code and the detailed results will be made available in an online supplement to this paper.

We calibrated parameters t and $I_{\max}$ of the tabu search with the irace package version 3.1 in GNU R [23]. For each objective, we ran irace with a budget of 1000 executions and a time limit of 10 minutes per run. We used a calibration data set consisting of 48 random instances generated with the same parameters and generator of data set DL. As [15] use tabu tenure values t relative to the number of districts. Table 2 shows the parameter ranges and best values found by irace, for each of the three objective functions.

In the following experiments we compare our method to existing approaches in the literature. For the p-median and p-center objectives we consider the exact algorithm of [34], which iteratively solves an IP model without connectivity constraints and adds cuts as unconnected solutions are encountered. Further, for the p-center objective we also consider the GRASP heuristic of [31]. For the diameter objective, we consider the multistart alternating tabu search heuristic of [15]. For the approaches of [34] and [15], we used the source codes provided by the authors. Because no source code for the algorithm of [31] was available, we reimplemented the method on the same platform as our own, and used parameters $\rho = 1.0, \beta = 0.5, A = \{0.1, 0.2, 0.3, 0.4, 0.5\}$ as given by [31] and $\lambda = 0.95$ as recommended by [28].

Each test was run with a time limit of 30 minutes. Since the method is stochastic, we report averages over 10 replications. For each experiment we group the results by number of units n. For the heuristic methods we report the average deviation of the compactness value relative to the best known value, in percent (C (%)), the number of multistart iterations ("iter."), the percentage of multistart iterations which resulted in a feasible solution ("fea. (%)"), and the time to find the best solution, in seconds ("t.t.b."). For the exact methods, we report the percentage of instances for which an optimal solution was found ("opt. (%)"), the running time in seconds ("time") and the number of IP models solved ("iter.").

6.3 Experiment 1: p-median objective

In this experiment we compare our approach to the exact method of [34] for the p-median objective. Table 3 shows the results.

Table 3: Comparison of our proposed method to the exact approach of [34] for the p-median objective.

n	Our algorithm				[34]		
	C (%)	t.t.b.	iter.	fea. (%)	opt. (%)	time	iter.
60	**0.00**	0	193,733	100	100	25	1.2
80	1.34	130	74,056	100	100	35	1.4
100	0.02	52	54,927	100	100	40	1.3
150	0.02	147	20,195	100	100	147	2.0
200	0.02	334	12,453	99	100	765	1.3
500	0.04	701	1,999	89	—	—	—
1,000	0.03	661	255	100	—	—	—
2,500	0.13	853	101	94	—	—	—
5,000	0.28	882	50	98	—	—	—
10,000	0.35	955	21	91	—	—	—
Avg.	0.22	471	35,779	97	100	202	1.4

Our heuristic found feasible solutions in 97% of the iterations, on average, considering all instance sizes. For instances of size $n \in \{60, 80, 150, 1000\}$ all iterations were feasible. Further, the heuristic reached optimality in all instances of size $n = 60$, and found solutions within 0.02% of optimality for $n \in \{100, 150, 200\}$. It obtained poor results in two particularly difficult instances of $n = 80$, which resulted in a higher deviation. As expected, the number of multistart iterations decreases as n increases.

The exact method of [34] reached optimality in all instances of size $n \leq 200$, and converges to a connected solution in less than 2 iterations, in average. This is most likely because the p-median objective reportedly tends to yield connected optimal solutions, given a relatively uniform planar topology. The exact solver timed out on all instances of $n > 200$ without finding a feasible upper bound. Although the proposed heuristic performed worse than the exact method, on average, it found the optimal solution in 84.6% of instances in data set DU, and was also able to consistently find feasible solutions for all instances of data sets DS, DT and DL, with n up to 10000.

6.4 Experiment 2: p-center objective

In this experiment we compare our approach to the GRASP heuristic of [31] and the exact method of [34] for the p-center objective. Table 4 shows the results.

Since the p-center objective is less related to connectivity compared to p-median, as reported by [34], their exact method generally requires more iterations to converge to a feasible solution. For $n \geq 200$, the exact method timed out in all cases without finding a feasible upper bound, whereas for $n = 100$ and $n = 150$, it found a solution in 90% and 50% of cases, respectively.

Considering instances solved optimally by the algorithm of [34] our heuristic found optimality in 95.4% of cases, and obtained an average relative deviation of 0.27% from the optimal solutions otherwise. For the remaining instances, the proposed heuristic obtained the best average objective values in all cases. On data set DA the heuristic of [31] performs a large number of iterations and thus obtains relatively good solutions, but has difficulty finding feasible solutions as n grows. This likely explains the poor solution quality for data sets DS, DT and DL, since the effectiveness of GRASP usually relies on having a large sample of feasible solutions.

Table 4: Comparison of our proposed method to the heuristic of [31] and the exact approach of [34], for the p-center objective.

n	Our algorithm				[31]				[34]		
	C (%)	t.t.b.	iter.	fea. (%)	C (%)	t.t.b.	iter.	fea. (%)	opt. (%)	time	iter
60	0.13	50	16,518	96	0.67	36	2,627,936	45	100	52	4.3
80	0.72	19	11,305	97	1.63	153	1,707,644	30	100	255	8.1
100	**0.00**	4	13,126	100	1.16	230	1,277,550	42	90	1,018	11.6
150	**0.06**	107	5,909	100	2.72	515	767,707	30	50	3,436	10.4
200	**0.00**	44	3,660	100	3.97	812	493,211	15	—	—	—
500	**0.31**	499	1,086	97	13.11	870	111,890	15	—	—	—
1,000	**0.79**	688	254	100	27.29	934	25,531	7	—	—	—
2,500	**1.68**	949	57	99	65.82	1,149	6,468	6	—	—	—
5,000	**1.62**	946	23	99	101.89	1,827	2,243	5	—	—	—
10,000	**1.76**	980	9	94	161.57	2,347	725	2	—	—	—
Avg.	**0.71**	429	5,195	98	37.98	887	702,091	20	85	1,190	8.6

Table 5: Comparison of our proposed method to the heuristic approach of [15] for the diameter objective.

n	Our algorithm				[15]			
	C (%)	t.t.b.	iter.	fea. (%)	C (%)	t.t.b.	iter.	fea. (%)
60	**0.00**	0	888,007	100	0.61	52	285,632	100
80	**0.00**	0	461,409	100	0.14	28	108,174	96
100	**0.00**	0	280,698	100	0.24	62	114,601	97
150	**0.00**	18	130,446	100	0.21	78	144,994	100
200	**0.03**	64	24,169	100	0.04	68	69,743	98
500	**0.07**	329	3,634	99	1.05	378	61,316	87
1,000	**0.33**	532	1,070	100	1.06	564	18,664	84
2,500	**1.82**	883	264	99	5.01	803	9,843	75
5,000	**2.91**	886	147	99	7.32	915	5,555	64
10,000	**5.31**	958	65	92	30.82	874	2,477	53
Avg.	**1.05**	367	178,991	99	4.65	382	82,100	85

Compared to the p-median objective, the p-center variant of our algorithm performs as little as 22.16 times fewer iterations in the same amount of time. This is mainly due to the differences in parameter $I_{\max}$ (see Table 2), which determines the amount of time spent on each tabu search. In fact, given equal t and $I_{\max}$ values, iterations which optimize p-center are usually faster, since balancing requires fewer binary search steps, in average. This happens because the number of possible p-center objective values is limited to $\binom{n}{2}$ (i.e., the maximum number of distinct node-node distances).

6.5 Experiment 3: diameter objective

In this experiment, we compare our approach with the heuristic method of [15] for the diameter objective. Table 5 shows the results.

Overall, our proposed heuristic produced solutions with lower average diameters in 38.7% of instances, and with equal diameters in 48.8% of instances. It also generally found solutions which are closer to the best known values, with an average relative deviation of 1.05%, compared to 4.65% for the algorithm of [15]. For the smaller data sets DU, DS and DT the proposed algorithm produced the best diameters in nearly all cases. For $n \leq 100$ it generates significantly more feasible solutions in the same amount of time, and finds the best known solutions almost immediately. As n grows, however, the total number of multistart iterations of our method decreases rapidly, which reduces the explored sample of the search space, causing comparatively higher diameters in some cases. This decrease in performance at higher values of n likely due to the increasing difficulty of finding balanced solutions.

In general, we found that the algorithm of [15] has more difficulties in balancing instances with high p/n ratios, especially as the number of units n grows. On the other hand, the performance of our method is less dependent on the number of districts p. Overall, 99% of the multistart iterations of our method were feasible, compared to 85% of [15]'s method.

Compared to the center-based objectives, the diameter approach is significantly faster per iteration. This is due to the high cost of computing the expected objective value of a neighbor, even with dynamic updates, which is $O(n/p)$ for p-median and p-center versus $O(\log n/p)$ for the diameter objective.

7 CONCLUSION

We have proposed a heuristic that can handle the three most common compactness measures in the literature: p-median, p-center and diameter, with little modification. It iteratively generates new solutions in a multistart fashion, and improves them by alternating between optimizing compactness and balance. Our main goal in this paper, however, was not to develop a state-of-the-art method which outperforms all others, but rather to establish a well-defined generic approach that can serve as a basis of comparison for future works considering one of the objectives above, as well as a baseline for new heuristics considering similar objectives or additional constraints.

The proposed method was compared to existing approaches in a widely-studied domain in commercial districting. For the p-center and p-median objectives, our heuristic yielded optimal solutions in almost all instances with less than 200 basic units in a fraction of the time, compared to a current exact approach. For the diameter objective, it improved upper bounds in 38.7% instances considered, on average, compared to a state-of-the-art heuristic approach.

ACKNOWLEDGMENTS

This research was supported by the Brazilian funding agencies CNPq (grant 420348/2016-6), FAPEMIG (grant TEC-APQ-02694-16) and by Google Research Latin America (grant 26568).

REFERENCES

[1] Paul K. Bergey, Cliff T. Ragsdale, and Mangesh Hoskote. 2003. A Simulated Annealing Genetic Algorithm for the Electrical Power Districting Problem. *Annals of Operations Research* 121, 1-4 (2003), 33–55.

[2] Steffen Borgwardt, Andreas Brieden, and Peter Gritzmann. 2014. Geometric Clustering for the Consolidation of Farmland and Woodland. *Mathematical Intelligencer* 36, 2 (2014), 37–44.

[3] Burcin Bozkaya, Erhan Erkut, and Gilbert Laporte. 2003. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research* 144, 1 (2003), 12–26.

[4] Jack Brimberg, Nenad Mladenović, and Dragan Urošević. 2015. Solving the maximally diverse grouping problem by skewed general variable neighborhood search. *Information Sciences* 295 (2015), 650–675.

[5] Alexander Butsch, Jörg Kalcsics, and Gilbert Laporte. 2014. Districting for Arc Routing. *INFORMS Journal on Computing* 26, October (2014), 809–824.

[6] M. Camacho-Collados, F. Liberatore, and J. M. Angulo. 2015. A multi-criteria Police Districting Problem for the efficient and effective design of patrol sector. *European Journal of Operational Research* 246, 2 (2015), 674–684.

[7] C. Chou, S. O. Kimbrough, J. Sullivan-Fedock, C. J. Woodard, and F. H. Murphy. 2012. Using Interactive Evolutionary Computation (IEC) with Validated Surrogate Fitness Functions for Redistricting. In *Genetic and Evolutionary Computation Conference - GECCO '12*. ACM, New York, 1071–1078.

[8] Steven J. D'Amico, Shoou-Jiun Wang, Rajan Batta, and Christopher M Rump. 2002. A simulated annealing approach to police district design. *Computers & Operations Research* 29, 6 (may 2002), 667–684.

[9] Juan C. Duque, Luc Anselin, and Sergio J. Rey. 2012. The Max-p-Regions Problem. *Journal of Regional Science* 52, 3 (aug 2012), 397–419.

[10] Mónica G. Elizondo-Amaya, Roger Z. Ríos-Mercado, and Juan A. Díaz. 2014. A dual bounding scheme for a territory design problem. *Computers and Operations Research* 44 (2014), 193–205.

[11] Erhan Erkut, Yilmaz Ülküsal, and Oktay Yeniçerioglu. 1994. A comparison of p-dispersion heuristics. *Computers and Operations Research* 21, 10 (1994), 1103–1113.

[12] Bernhard Fleischmann and Jannis N. Paraschis. 1988. Solving a large scale districting problem: A case report. *Computers & Operations Research* 15, 6 (1988), 521–533.

[13] R. S. Garfinkel and G. L. Nemhauser. 1970. Optimal Political Districting by Implicit Enumeration Techniques. *Management Science* 16, 8 (1970), B–495–B–508.

[14] Alex Gliesch, Marcus Ritt, and Mayron C. O. Moreira. 2017. A genetic algorithm for fair land allocation. In *Genetic and Evolutionary Computation Conference - GECCO '17*. 793–800.

[15] Alex Gliesch, Marcus Ritt, and Mayron C. O. Moreira. 2018. A Multistart Alternating Tabu Search for Commercial Districting. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10782 LNCS. 158–173.

[16] Fred Glover. 1986. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* 13, 5 (jan 1986), 533–549.

[17] Saïd Hanafi, Arnaud Freville, and Polo Vaca. 1999. Municipal Solid Waste Collection: An Effective Data Structure For Solving The Sectorization Problem With Local Search Methods. *INFOR: Information Systems and Operational Research* 37, 3 (1999), 236–254.

[18] S. W. Hess, J. B. Weaver, H. J. Siegfeldt, J. N. Whelan, and P. A. Zitlau. 1965. Nonpartisan Political Redistricting by Computer. *Operations Research* 13, 6 (1965), 998–1006.

[19] Diana L. Huerta-Muñoz, Roger Z. Ríos-Mercado, and Rubén Ruiz. 2017. An iterated greedy heuristic for a market segmentation problem with multiple attributes. *European Journal of Operational Research* 261, 1 (2017), 75–87.

[20] Jörg Kalcsics. 2015. Districting Problems. In *Location Science*. Springer International Publishing, Cham, 595–622.

[21] Douglas M. King, Sheldon H. Jacobson, and Edward C. Sewell. 2017. The geograph in practice: creating United States Congressional Districts from census blocks. *Computational Optimization and Applications* (2017), 1–25.

[22] Hongtao Lei, Gilbert Laporte, Yajie Liu, and Tao Zhang. 2015. Dynamic design of sales territories. *Computers and Operations Research* 56 (2015), 84–92.

[23] Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Leslie Pérez Cáceres, Mauro Birattari, and Thomas Stützle. 2016. The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives* 3 (2016), 43–58.

[24] Anuj Mehrotra, Ellis L. Johnson, and George L. Nemhauser. 1998. An Optimization Based Heuristic for Political Districting. *Management Science* 44, 8 (1998), 1100–1114.

[25] L. Muyldermans, D. Cattrysse, D. Van Oudheusden, and T. Lotan. 2002. Districting for salt spreading operations. *European Journal of Operational Research* 139, 3 (2002), 521–532.

[26] Federica Ricca, Andrea Scozzari, and Bruno Simeone. 2013. Political Districting: From classical models to recent approaches. *Annals of Operations Research* 204, 1 (2013), 271–299.

[27] Federica Ricca and Bruno Simeone. 2008. Local search algorithms for political districting. *European Journal of Operational Research* 189, 3 (2008), 1409–1426.

[28] Roger Z. Ríos-Mercado. 2016. Assessing a Metaheuristic for Large-Scale Commercial Districting. *Cybernetics and Systems* 47, 4 (2016), 321–338.

[29] Roger Z. Ríos-Mercado. 2017. *Tabu Search with Strategic Oscillation for Improving Recollection Assignment Plans of Waste Electric and Electronic Equipment*. Technical Report. Universidad Autónoma de Nuevo León.

[30] Roger Z. Ríos-Mercado and Hugo Jair Escalante. 2016. GRASP with path relinking for commercial districting. *Expert Systems with Applications* 44, September 2015 (2016), 102–113.

[31] Roger Z. Ríos-Mercado and Elena Fernández. 2009. A reactive GRASP for a commercial territory design problem with multiple balancing requirements. *Computers and Operations Research* 36, 3 (2009), 755–776.

[32] Roger Z. Ríos-Mercado and J. Fabián López-Pérez. 2012. Commercial territory design planning with realignment and disjoint assignment requirements. *Omega (United Kingdom)* 41, 3 (2012), 525–535.

[33] Roger Z. Ríos-Mercado and Juan C. Salazar-Acosta. 2011. A GRASP with Strategic Oscillation for a Commercial Territory Design Problem with a Routing Budget Constraint. In *10th International Conference on Artificial Intelligence: Advances in Soft Computing*. Vol. 7095 LNAI. 307–318.

[34] María Angélica Salazar-Aguilar, Roger Z. Ríos-Mercado, and Mauricio Cabrera-Ríos. 2011. New Models for Commercial Territory Design. *Networks and Spatial Economics* 11, 3 (2011), 487–507.

[35] M. Angélica Salazar-Aguilar, Roger Z. Ríos-Mercado, and José Luis González-Velarde. 2011. A bi-objective programming model for designing compact and balanced territories in commercial districting. *Transportation Research Part C: Emerging Technologies* 19, 5 (2011), 885–895.

[36] M. Angélica Salazar-Aguilar, Roger Z. Ríos-Mercado, and José Luis González-Velarde. 2013. GRASP strategies for a bi-objective commercial territory design problem. *Journal of Heuristics* 19, 2 (2013), 179–200.

[37] M. Angélica Salazar-Aguilar, Roger Z. Ríos-Mercado, José L. González-Velarde, and Julián Molina. 2012. Multiobjective scatter search for a commercial territory design problem. *Annals of Operations Research* 199, 1 (oct 2012), 343–360.

[38] J. Ángel Segura-Ramiro, Roger Z. Ríos-Mercado, Ada M. Álvarez-Socarrás, and Karim de Alba Romenus. 2007. A Location-Allocation Heuristic for a Territory Design Problem in a Beverage Distribution Firm. *Proceedings of the 12th Annual International Conference on Industrial Engineering - Theory, Applications and Practice* November (2007), 428–434.

[39] Michael Ian Shamos. 1978. *Computational Geometry*. Ph.D. Dissertation. Yale University.

[40] Maria Teresinha Arns Steiner, Dilip Datta, Pedro José Steiner Neto, Cassius Tadeu Scarpin, and José Rui Figueira. 2015. Multi-objective optimization in partitioning the healthcare system of Parana state in Brazil. *Omega (United Kingdom)* 52 (2015), 53–64.

[41] R. Endre Tarjan. 1974. A note on finding the bridges of a graph. *Inform. Process. Lett.* 2, 6 (1974), 160–161.

[42] Seda Yanik, Özge Sürer, and Basar Öztaysi. 2016. Designing sustainable energy regions using genetic algorithms and location-allocation approach. *Energy* 97 (2016), 161–172.