CMP189 Geometric Algorithms – final project Heuristic polygonal approximation of discrete land subdivisions

Alex Gliesch[1]

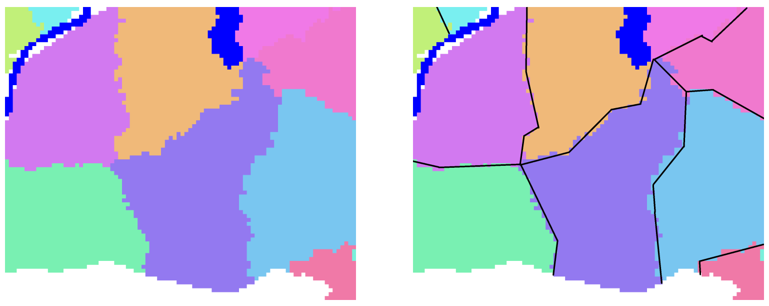
Federal University of Rio Grande do Sul, Porto Alegre, Brazil
alex.gliesch@inf.ufrgs.br

1 Introduction

The territorial organization in agrarian reform projects and environmental planning problem (PROTERRA, acronym for the name in Portuguese) aims at subdividing a large parcel of agrarian land, which previously had a single owner, into many small lots to be designated to families. This problem is difficult, since several legal and ethical constraints must be considered. For example, lots must be divided more or less equally in terms of soil quality and access to road networks and water resources, and the subdivisions must respect natural boundaries such as preservation areas. Most methods to solve this problem have been manual, time-consuming and often flawed, usually producing rectangular lots which, although simple and compact to define, fail to consider these constraints.

Recently, Gliesch et al. (2017) proposed a genetic algorithm to find fair land subdivisions for PROTERRA by computer. They use a discrete approach in which the input map is represented as a regular grid of equal-sized cells, each representing a river, land, or a natural preservation area. The problem then is modelled as a districting problem Kalcsics 2015 where each cell (or unit) is to be assigned to a different lot (or district). Their method generates initial solutions by a greedy constructive algorithm following a location-allocation approach Kalcsics et al. 2005, and uses novel genetic operators which are based on deconstructing part of the solution and reconstructing it again. Experiments show that their methods are able to produce feasible solutions for all 5 of the real-world instances considered, which are much better balanced than the previous manual approaches. Further, because their algorithm is mainly based on a constructive heuristic which expands lots similarly to a breadth-first search, their solutions tend to be somewhat compact, although compactness was not something explicitly considered by the paper.

However, because a discrete approach is considered, the solutions produced by Gliesch et al. (2017)'s method usually have lots with "jagged" or irregular borders, comprised of a very large number of small edges. Figure 1a illustrates this issue. This is highly undesirable in practice, since it makes it difficult to define lot borders in legal documents or to demarcate the boundaries physically (e.g. building a fence that separates the lots). A more desirable subdivision would



(a) An unacceptable subdivision, full of artifacts.

(b) A likely more acceptable approximation of the subdivision.

Fig. 1: Motivation behind our desire to approximate discrite subdivisions.

be one that achieves more or less the same balancing effect, but with a minimal number of edges per lot, as illustrated by Figure 1b.

The main motive for this paper is to develop a method to simplify a discrete grid-based land subdivision obtained by a computerized method such as the one of Gliesch et al. (2017). To do this, we aim to define lots as simple polygons in a continuous environment. These polygons should have as few edges as possible, and closely resemble the original lots in area.

In more specific terms, in this paper we consider two problem variants. The first one sets a fixed maximum number of edges per polygon, and aims at defining such polygons in a way to minimize the difference in area between the continuous and the discrete lots. The second one sets a maximum deviation threshold of area of a lot's discrete and continuous solution, and aims at minimizing the number of edges per lot.

2 Definitions

We are given an integer rectangular grid $U = R \mathbin{\dot\cup} P \mathbin{\dot\cup} L$ of size $n \times m$, where R, P and L represent river, natural preserve, and land cells, respectively. For each unit $u \in L$ we are given the lot $l(u) \in [k]$ that u is assigned to. We assume that all units assigned to a given lot are connected on a standard grid 4-neighborhood. Further, we are given the following parameters: a maximum desired number of edges per lot $E > 2$, a maximum lot area deviation threshold $\tau \geq 0$ from the original solution, a minimum polygon internal angle $\theta \in [0, 2\pi]$ and a minimum polygon edge length $L > 0$. Note that, differently from Gliesch et al. (2017), in this paper we do not consider land productivity or river access, but focus solely on area. Further, we do not discriminate between feasible versus infeasible, or balanced versus imbalanced subdivisions with respect to the constraints considered by Gliesch et al. (2017): we accept subdivisions as they are given, as long as there is a complete mapping of L into contiguous lots.

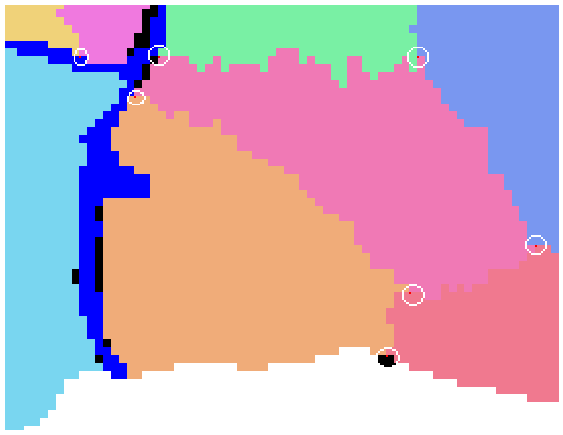


Fig. 2: The pivot locations are highlighted in white.

We define a *polygon mapping* $\mathscr{P}$ as a set of simple polygons $\{P_1, \ldots, P_k\}$, one for each lot, where $P_i = (v_{i1}, \ldots, v_{iz})$ is a list of vertices in clockwise order. We say that an edge of a polygon $P \in \mathscr{P}$ is *man-made* if it also appears in another polygon $Q \neq P$, otherwise we say that the edge is *natural.* A polygon mapping $\mathscr{P}$ is said to be valid if it satisfies the following criteria:

1. Completeness: the space occupied by L is equal to the space occupied by all $P \in \mathscr{P}$ combined.
2. Edge length: the minimum length of a man-made edge for each $P \in \mathscr{P}$ is greater than or equal to L.
3. Angle: the minimum internal angle between two man-made edges in each $P \in \mathscr{P}$ is greater than or equal to θ.
4. Number of edges: each $P \in \mathscr{P}$ has at most E man-made edges.
5. Deviation from original: the relative deviation between the area of each $P \in \mathscr{P}$ and the area of the respective lot in the discrete solution is at most τ.

Our approach focuses on generating mappings that preserve original neighboring relations between lots. To this end, we define a set of *pivot* locations, which are map locations that simultaneously sit on the border of more than two lots, or of two lots and either a river or a natural preservation area. Figure 2 illustrates pivots. The pivots can be easily identified by iterating over all grid intersections in U. We further define a set of *paths* C, each of which contains a list of grid intersections on the boundary between two lots. Paths are identified by a depth-first search starting at a pivot on said boundary and ending as soon as another pivot is encountered. Consequently, all paths start and end at a pivot. Note, however, that a given pair of lots may have more than one path between them. We refer to the map locations in a path as *nodes.* We define the set of all nodes as N.

A solution is a mapping $S : N \to \{0, 1\}$. A node $n \in N$ is said to be *enabled* in S if $S(n) = 1$, otherwise it is *disabled.* A polygon mapping $\mathscr{P}(S)$ can be obtained from a solution by connecting consecutive enabled nodes on each path with man-made edges, and connecting the boundaries between lots and natural obstacles with natural edges. Note that, if all pivots are enabled, the mapping will be complete. Further, we define $d(S, l)$ to be the relative deviation of the area of lot l in $\mathscr{P}(S)$ to its original area in the discrete solution, $e(S, l)$ the number of man-made edges of lot l in $\mathscr{P}(S)$, $ds(S)$ the list of lot area deviations in S in decreasing order and finally $es(S)$ the list of man-made lot edges in decreasing order.

We say that a solution is *feasible* if all pivots are enabled and its respective polygon mapping is valid. If a fixed maximum number of edges E is set, we use $\tau = -\infty$ and the goal is to find a feasible solution that minimizes $ds(S)$. If a fixed maximum deviation τ is set, we use $E = \infty$ and the goal is to find a feasible solution that minimizes $es(S)$.

3 Proposed algorithm

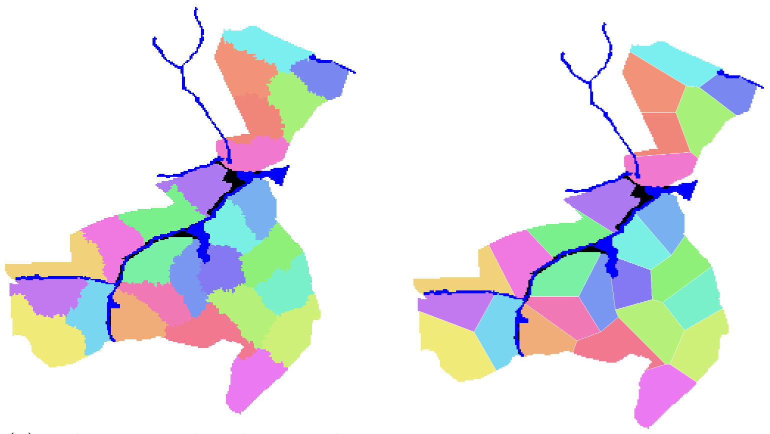
We propose a simple deterministic heuristic composed of a greedy constructive procedure followed by a local search. Starting from an initial solution with as few man-made edges as possible, a constructive heuristic repeatedly splits a greedily-selected edge in two, until a termination criterion is reached. Then, a simple local search procedure is applied, which may continue splitting edges or merging two existing edges into one.

3.1 Constructive heuristic

Initial solutions are obtained by having only the pivot nodes enabled. Since only the pivot nodes are selected, the initial solution has the lowest possible number of edges. Figure 3 illustrates the initial solution for a subdivision of the real-world instance “Veredas”.

Then, we expand solutions as follows. First, we obtain an initial solution S, and mark all lots as “open”. Then, we repeatedly select the open lot l with highest $d(S, l)$ and the disabled node n in some path incident to l which, when enabled, yields a feasible solution S' with the smallest $(ds(S'), es(S'))$, compared lexicographically. If no n yields a feasible solution (because it violates either the edge length, angle and number of edges criteria described in Section 2) or if the best S' is worse than the incumbent, we mark l as “closed” and continue on to the next lot; otherwise, we accept the move and enable n. We stop when all lots are marked as closed, when $d(S, l) <= \tau$, or when a time limit is reached. The algorithm is detailed in Algorithm 1.

In practice, we attempt to produce as few man-made edges per lot as possible by greedily selecting the node which produces the steepest decrease of $(ds(S), es(S))$, since we expect that this will lead to fewer nodes being enabled until a local minimum is found. In the case of a fixed maximum edges per lot, we close lots when their number of edges reaches L, since no possible node to be enabled will yield a feasible solution. In the case of a fixed maximum area deviation, the algorithm stops as soon as the largest deviation is less than τ.



(a) The original solution obtained by the algorithm of Gliesch et al. (2017).

(b) The initial constructive solution with only the pivots enabled.

Fig. 3: Original solution (left) and initial approximation by pivots (right) of the instance "Veredas".

3.2 Local search

The constructive heuristic is followed by a simple local search, which allows both the enabling and disabling of nodes. The main idea here is that, by disabling and then enabling some nodes, we can improve the location of edge splits, or undo an edge split that is not necessary to achieve the current ds.

The local search uses a different objective function depending on whether the largest lot area deviation exceeds τ or not. It iteratively selects the node $n \in N$ which, when toggled, minimizes $(ds(S), es(S))$ if $min_{l\in[k]} d(S,l) \geq \tau$, or $(es(S), ds(S))$ otherwise. If toggling n improves the incumbent, we accept the movement and continue, otherwise we end the search. Algorithm 2 shows the procedure in more detail.

3.3 Implementation details

We use discrete approximation to compute the polygon areas. That is, given a discretization factor f, we represent polygon mappings on a regular grid of size $nf \times mf$, where each cell is mapped to a polygon or an edge. The polygon edges are drawn using Bresenham's line algorithm, and the areas are computed by performing, for each lot, a breadth-first search that starts from an *anchor* cell and uses the polygon edges as stopping points. Anchors are cells that will certainly belong to the same lot in all solutions, and are pre-computed by drawing Bresenham-lines between all pairs of nodes in each path, and then selecting any node in the largest connected component of each lot. Figure 4 illustrates this process.

Algorithm 1 constructive heuristic

```
 1: $closed(l) \leftarrow$false $\forall l \in [k]$
 2: $S \leftarrow$ initialSolution()
 3: **repeat**
 4:     $l \leftarrow \arg\max\{d(S,l) \mid \neg closed(l)\}$
 5:     $S' \leftarrow S$
 6:     **for** $n \in$ nodesAdjacentTo($l$) **do**
 7:         **if** $\neg$ isEnabled($S$, $n$) **then**
 8:             enable($S$, $n$)
 9:             **if** isFeasible($S$) **and** $(ds(S), es(S)) < (ds(S'), es(S'))$ **then**
10:                 $S' \leftarrow S$
11:             disable($S$, $n$)
12:     **if** $S \neq S'$ **then** $S \leftarrow S'$
13:     **else**
14:         $closed(l) \leftarrow$true
15: **until** $closed(l)$ $\forall l \in [k]$
16: **return** $S$
```

We have opted for this discrete approach since it is more easily extendable (e.g. it can be adapted to consider soil productivity or other criteria instead of area), it is easy to code, and the difference in performance was negligible compared to a geometric approach. With a large enough discretization factor, the difference in precision also becomes negligible. In our implementation we use $f = 6$, as we have found it to be a good compromise between precision and performance.

4 Computational experiments

4.1 Experimental setup

We have coded our algorithms in C++, and compiled them with GCC 7.2.0 with optimization `-O3`. All experiments were performed on a PC with a 4-core Intel i7-6700 processor and 16 GB of main memory, and were limited to 30 minutes of running time. For each of the 5 real-world scenarios available from Gliesch et al. (2017) ("Veredas", "Belo Vale", "Olhos D'água", "Fortaleza" and "Iucatã"), we have generated 10 subdivisions using the genetic algorithm of Gliesch et al. (2017) with default parameters and 2 minutes of running time, for a total of 50 instances. The instance sizes range from 300×300 to 449×250, and have between 24 and 40 lots.

In the following, we report on experiments that evaluate the two variants of the algorithm: with a fixed maximum number of edges E, and with a fixed maximum relative deviation τ. Because the algorithm proposed is deterministic, only one replication of each configuration was performed per instance. For these experiments, we have fixed $L = 4$ and $\theta = \pi/3$, as they seem to produce the most visually acceptable solutions.

Algorithm 2 local search

```
1: $ended$ ←false
2: repeat
3:     $S' \leftarrow S$
4:     for $n \in N$ do
5:         toggle($S$,$n$)
6:         if isFeasible($S$) then
7:             if $(max(ds(S)) > \tau$ and $(ds(S), es(S)) < (ds(S'), es(S')))$ or
8:               $(max(ds(S)) \leq \tau$ and $(es(S), ds(S)) < (es(S'), ds(S')))$ then
9:                 $S' \leftarrow S$
10:        toggle($S$,$n$)
11:    if $S \neq S'$ then $S \leftarrow S'$
12:    else
13:        $ended$ ←true
14: until $ended$
15: return $S$
```

Table 1: Maximum (average) deviation in area, in %, given a fixed maximum number of edges per lot E.

Instance \ E	4	6	8	10	12	14	16
Belo Vale	30.9 (7.1)	30.5 (6.2)	24.5 (4.5)	17.9 (3.0)	13.2 (1.9)	7.9 (1.3)	5.7 (1.1)
Fortaleza	31.9 (7.1)	12.7 (3.0)	5.3 (1.3)	2.6 (1.0)	2.4 (1.0)	2.4 (0.9)	2.4 (0.9)
Iucatã	35.3 (7.2)	17.0 (3.0)	9.7 (1.5)	3.0 (0.8)	2.3 (0.7)	2.2 (0.7)	2.1 (0.7)
Olhos D'Água	29.9 (5.9)	14.5 (2.9)	7.4 (1.3)	3.6 (0.8)	3.2 (0.7)	1.9 (0.5)	1.3 (0.4)
Veredas	19.4 (5.0)	11.0 (2.3)	3.9 (1.0)	2.9 (0.8)	2.8 (0.7)	2.4 (0.7)	2.4 (0.7)
Average	29.5 (6.5)	17.2 (3.4)	10.2 (1.9)	6.0 (1.3)	4.8 (1.0)	3.4 (0.8)	2.8 (0.8)

4.2 Experiment 1: fixed number of edges

This experiment aims at evaluating the effectiveness of the algorithm at minimzing the relative deviation in area from the original solution per lot, when we are given a maximum number of edges $E \in \{4, 6, 8, 10, 12, 14, 16\}$. Table 1 shows the maximum (average) area deviation per lot, in %, for each instance and E, averaged over each of the 10 subdivisions.

We can see that the algorithm has major difficulties approximating the area of some lots when $E = 4$, averaging a maximum of 30% deviation. As E increases, however, we are able to achieve much better approximations. The improvement rate decays significantly after $E = 10$ for all instances except "Belo Vale", achieving a maximum deviation between 3.2% and 2.4%. This is expected, since as the number of edges grows, the addition or removal of a new one incurs a smaller difference. Furthermore, we can observe that the average deviation in area per lot is already quite low with $E = 6$, averaging a 3.4% deviation per lot. This indicates that it is typically not difficult to achieve a good approximation

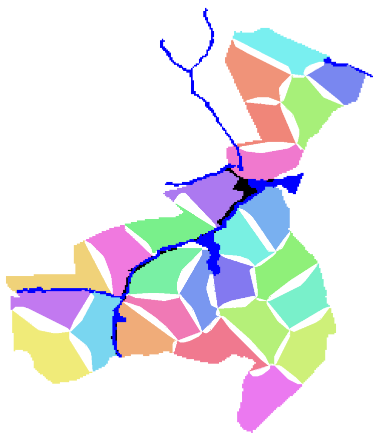


Fig. 4: The white regions on the lot boundaries represent regions whose assignment may change depending on which nodes are selected, and the colored regions represent the anchors, i.e., regions which always have the same lot in all solutions.

($< 5\%$) with few edges, and that the maximum deviations tend to appear in a few exceptional lots that are hard to approximate.

The instance "Belo Vale" appears to be the hardest one to achieve good approximations, being worse than 5% even with 16 edges per lot. This is because it has a thin pattern of natural preserves spread vertically over the map, as shown in Figure 5, and thus requires a very large number of edges per lot to achieve a good approximations. In the real-world this is a road, which, for simplicity, has been modelled as a natural preserve since it represents an area that cannot be allocated.

Figure 6 displays example solutions for the instance "Fortaleza" with $E = 4$, $E = 8$ and $E = 16$.

4.3 Experiment 2: fixed maximum deviation

This experiment aims at evaluating the effectiveness of the algorithm at minimzing the number of edges per lot when we are given a maximum relative deviation in lot area $\tau \in \{0.01, 0.025, 0.5, 0.075, 0.1, 0.125\}$. Table 2 shows the maximum (average) number of edges per lot for each instance and τ, averaged over each of the 10 subdivisions. For instances "Belo Vale", "Fortaleza" and "Iucatã", the algorithm was not able to find a subdivision whose maximum relative deviation is less than 1%.

We can see that, as τ grows, the maximum number of edges necessary to achieve it decreases, as expected, since a coarser approximation is easy to achieve with fewer edges. Further, the average number of edges per lot is typically much lower than the maximum, averaging less than 5 for $\tau > 5\%$, for example.

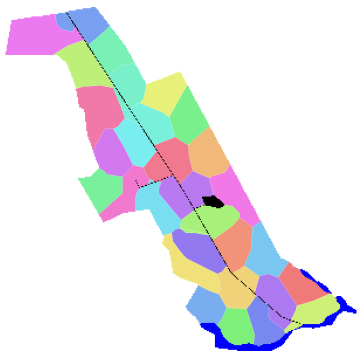


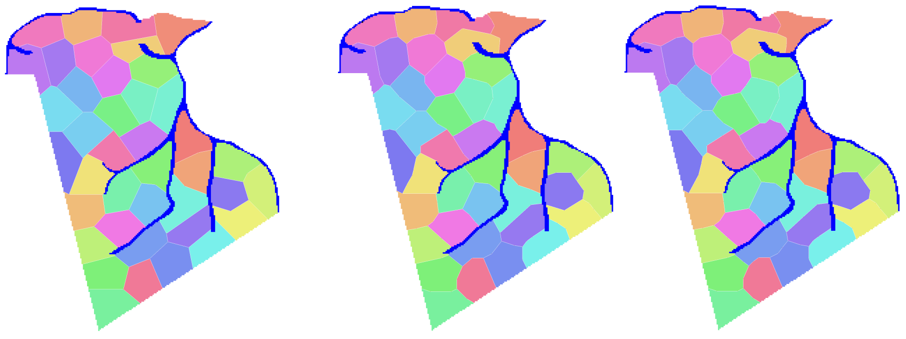
Fig. 5: Instance "Belo Vale" has a road spread vertically over the map, which induces a large number of edges per lot.

Table 2: Maximum (average) number of edges per lot, given a fixed maximum deviation τ.

Instance \ $\tau(\%)$	1.0	2.5	5.0	7.5	10.0	12.5
Belo Vale	—	17.6 (8.4)	16.7 (7.6)	16.3 (7.3)	16.2 (7.1)	15.9 (7.0)
Fortaleza	—	10.1 (5.7)	8.3 (4.9)	8.0 (4.5)	7.6 (4.3)	7.6 (4.1)
Iucatã	—	10.8 (4.7)	9.0 (4.0)	8.6 (3.7)	7.9 (3.5)	7.78 (3.4)
Olhos D'Água	12.4 (5.5)	11.2 (4.7)	10.2 (4.2)	9.9 (3.9)	9.4 (3.8)	9.3 (3.7)
Veredas	10.0 (5.7)	9.28 (4.8)	8.0 (4.1)	7.2 (3.6)	6.8 (3.4)	6.5 (3.3)
Average	15.6 (8.1)	12.5 (6.1)	10.6 (4.7)	10.1 (4.7)	9.6 (4.4)	9.4 (4.3)

An interesting note is that this experiment performed significantly better than the previous one, in some instances: for Veredas, for example, with at most 10 edges we obtained an approximation of 1%, whereas in the previous experiment the best maximum deviation for "Veredas" was 2.4% with 16 edges. The same occurs for instances "Olhos D'Água" and "Iucatã". This is likely because, in this verison, the constructive heuristic typically ends earlier (because a maximum deviation of τ was achieved), and the local search that follows directly is able to reduce the number of edges further, since it allows both enabling and disabling nodes. As before, the instance "Belo Vale" seems to be the hardest one.

Figure 7 displays example solutions for the instance "Veredas" (the original subdivision is shown in Figure 3) with $\tau = 1\%$, $\tau = 5\%$ and $\tau = 10\%$. Looking closely, one can see that the differences are subtle.



(a) $E = 4$, maximum deviation 28.9% (average 6.2%).

(b) $E = 8$, maximum deviation 1.9% (average 0.9%).

(c) $E = 16$, maximum deviation 1.4% (average 0.9%).

Fig. 6: Example solutions for the instance "Fortaleza" with differing fixed number of edges.

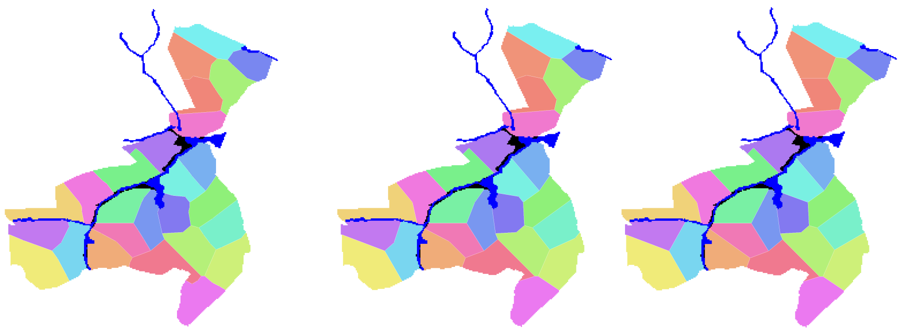
5 Conclusion

This paper addressed the problem of approximating discrete land subdivisons, which are defined by a mapping of cells on a regular rectangular grid to lots, through a set of simple polygons, one for each lot. We have considered two variants: one which fixes the maximum number of edges and aims at obtaining a solution as close to the original one as possible, and one which fixes a maximum relative deviation to the original and attemps to minimize the number of polygon edges. We have proposed a simple heuristic which obtains initial solutions by connecting pivot locations and repeatedly splitting edges in two, followed by a local search which both splits and merges edges. Experiments with real-world instances show that our algorithm is able to produce good approximations of the original solutions, achieving a maximum relative deviation of 2.5% with as most as 11 edges per lot, in most instances. Furthermore, by setting bounds for the minimum internal angle and edge length of each lot, the approximations produced are visually pleasing and much closer to ones that can be applied to practice.

References

Gliesch, A., M. Ritt, and M. C. O. Moreira (2017). "A genetic algorithm for fair land allocation". In: *Genetic and Evolutionary Computation Conference - GECCO '17*. New York, New York, USA: ACM Press, pp. 793–800.

Kalcsics, J. (2015). "Districting Problems". In: *Location Science*. Cham: Springer International Publishing, pp. 595–622.



(a) $\tau = 1\%$, at most 13 edges (7.46 on average).

(b) $\tau = 5\%$, at most 8 edges (3.84 on average).

(c) $\tau = 10\%$, at most 8 edges (3.30 on average).

Fig. 7: Example solutions for the instance “Veredas” with differing fixed maximum relative deviations.

Kalcsics, J., S. Nickel, and M. Schröder (2005). “Towards a unified territorial design approach - applications, algorithms and GIS integration”. In: *Top* 13.1, pp. 1–56.