ON ENUMERATING ALL FEASIBLE SOLUTIONS TO POLYGON AGGREGATION PROBLEMS

Michael F. Goodchild
and
Catherine M. Hosage
Department of Geography
The University of Western Ontario
London, Canada

ABSTRACT

A set of simple rules allows the problem of aggregating n zones into m regions to be represented in the form of a combinatorial tree. A branch and bound algorithm is described for searching this tree given a suitable structuring of the zone boundary network as a set of edge records with pointers. The performance of the algorithm on highly bounded electoral districting problems is compared to that of other methods in the literature.

INTRODUCTION

Consider a set of n contiguous zones, defined spatially by a boundary network of edges and vertices. The problem considered is that of exhaustively partitioning the n zones into a prescribed number of regions or electoral districts. We define an m-partition as a partitioning into exactly m regions. In general the number of possible m-partitions, P(n,m), depends on the specific arrangement of zones and simple expressions exist only for a few special cases. Cliff and Haggett (1) give expressions for two highly symmetrical arrangements, first with every zone in contact with every other zone, and second, with all n zones arranged contiguously in a ring so that each one is adjacent to exactly two neighbours.

In the electoral districting case, and in other forms of region-building as well, there will be constraints which further limit the number of feasible partitions. Often the set will be constrained by some combination of upper and lower bounds on the total population of each region, or by bounds on total area or shape. For the purpose of this paper the discussion will be restricted to population bounds, but others could be added by simple extension. We write the number of feasible partitions which satisfy the constraints as $F(n,m) \leqslant P(n,m)$.

The problem considered in this paper is that of finding all F(n,m) feasible solutions efficiently. Perhaps the earliest attempt at a complete enumeration was by Garfinckel and Nemhauser (2), who used a two-stage algorithm. They assumed that each region would be constrained by upper bounds on population and on a dimensionless measure of compactness. In addition an exclusion matrix would be provided to prevent certain pairs of zones from being members of the same district. This matrix can clearly have a marked effect on the efficiency of the algorithm, but it would be difficult to generate one in practice without running the risk of excluding otherwise feasible solutions.

In the first stage, Garfinckel and Nemhauser's algorithm generates all possible single regions. These are filed, and then the second stage searches for combinations which allocate each zone exactly once, and therefore constitute feasible aggregations. Both stages can be programmed efficiently as branch and bound algorithms. In the political districting application an objective function of optimum compactness was applied to select a single best solution, provided at least one feasible solution was found to exist. The objective function can of course be used as a further upper bound on the second stage of the algorithm.

More recent literature in electoral districting has argued that no single objective function is appropriate, and that it is more useful to produce all F(n,m) feasible plans, or at least a sample of them, so that the final selection can be made using less tangible or more complex criteria. A recent paper by Rossiter and Johnston (3), which presents an algorithm for enumerating all feasible plans, provided much of the motivation for the research described here.

Their algorithm begins by randomly choosing m core zones. In a modification to the basic method, these can be selected so that one core is drawn randomly from each of m subsets identified by the user prior to the analysis. The algorithm then proceeds by randomly adding contiguous but unassigned zones to the cores until all zones have been allocated. Upper bounds on population and area can be imposed as the regions are built, but lower bounds and constraints on shape can be imposed only on completion of all regions. A similar approach has been described by Minciardi et al. (4).

A single run of the algorithm may or may not produce a feasible solution, depending on the constraints. If one is produced, it can be regarded as a random sample from among the F(n,m) feasible alternatives, but there is no reason to suppose that all F(n,m) are equi-probable. To be reasonably confident that all F(n,m) have been produced it is necessary to run the algorithm a very large number of times, checking after each successful run whether the solution has been found before. However one can never be certain that all solutions have been found, particularly if the modified form has been used in which cores are drawn from predefined subsets.

Rossiter and Johnston report considerable computational experience with the algorithm. For example, 200,000 runs of a districting problem in Sheffied with n = 27, m = 6 and with upper and lower bounds on population produced 15,937 distinct solutions, the last new solution occuring in run number 199,001. In total 6,000 cpu seconds were required on an ICL machine.

The algorithm described below structures the problem in the form of a combinatorial tree, and uses a branch and bound method to enumerate all feasible solutions. As such it can be guaranteed always to produce all feasible solutions. Furthermore the algorithm becomes increasingly efficient as the number of feasible alternatives is reduced through the use of tighter bounds on each region's attributes. It relies on a data structure which gives a more complete description of the boundary network of the n zones than is given by the adjacency matrix used by Rossiter and Johnston and by Garfinckel and Nemhauser.

ALGORITHM

Consider the boundary network in Figure 1. We assume that all vertices are 3-valent, and in practice ensure this by pre-processing the boundary network. Any 2-valent vertices are deleted and the incident edges merged. Any p-valent vertices, $p \geqslant 4$, are replaced by p 3-valent vertices and p zero-length arcs. The hole produced is given a label which presents it becoming part of any region. By implication, two zones are adjacent only if they share an edge of the boundary network of non-zero length.

Define the area exterior to the n zones as the outside world, or zone 0. We first select an edge of the boundary network adjacent to zone 0, and label it the root edge. The zone which it encloses is labelled the root zone, and the vertex reached by traversing the root edge anticlockwise to the root zone is defined as the root vertex (Figure 1).

Beginning at the root vertex and traversing the root edge, consider all possile circuits in the boundary network which return to the root vertex. We follow the conventional definition of a circuit as a continuous path which returns to the origin after visiting each vertex at most once and traversing each edge at most once. We represent the set of all possible paths as a tree rooted at the root vertex, and with the root vertex at each leaf node. The branches of the tree represent the options of turning left and right at each vertex. The left/right sense can be maintained in drawing the tree, as shown in Figure 2, where vertex and edge labels are as in Figure 1.

Note that in some cases the tree fails to branch at a vertex, because one option leads to a vertex which has already been visited. The tree can also be truncated wherever a turn would involve traversing an edge with the exterior, zone 0, to the right.

Each leaf node can be labelled with the set of zones enclosed by the circuit, as shown in Figure 2. These include all zones lying to the right of traversed edges, although in larger problems a circuit can enclose zones which are not actually adjacent to the circuit itself.

If regions must be bounded by a single circuit, and if regions may not enclose other regions, then the set of all circuits enumerated in Figure 2 is the set of all possible regions containing the root zone. Suppose now that we wish to enumerate all 2-partitions. The second region is defined simply as the set of zones remaining at each leaf of Figure 2. In the case of (1234) there can be no second region as no zones remain, and in the case of (13) the remaining zones are not contiguous, so we conclude that P(4,2) = 6. For m > 2, the process is iterated. The remaining zones at each leaf are searched for a

Figure 1. Boundary Network Showing Root Edge, Root Vertex and Root Zone.

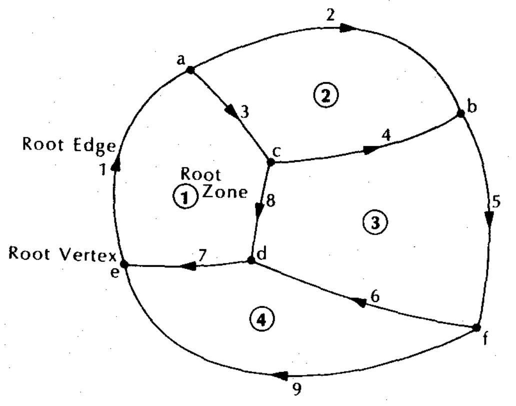
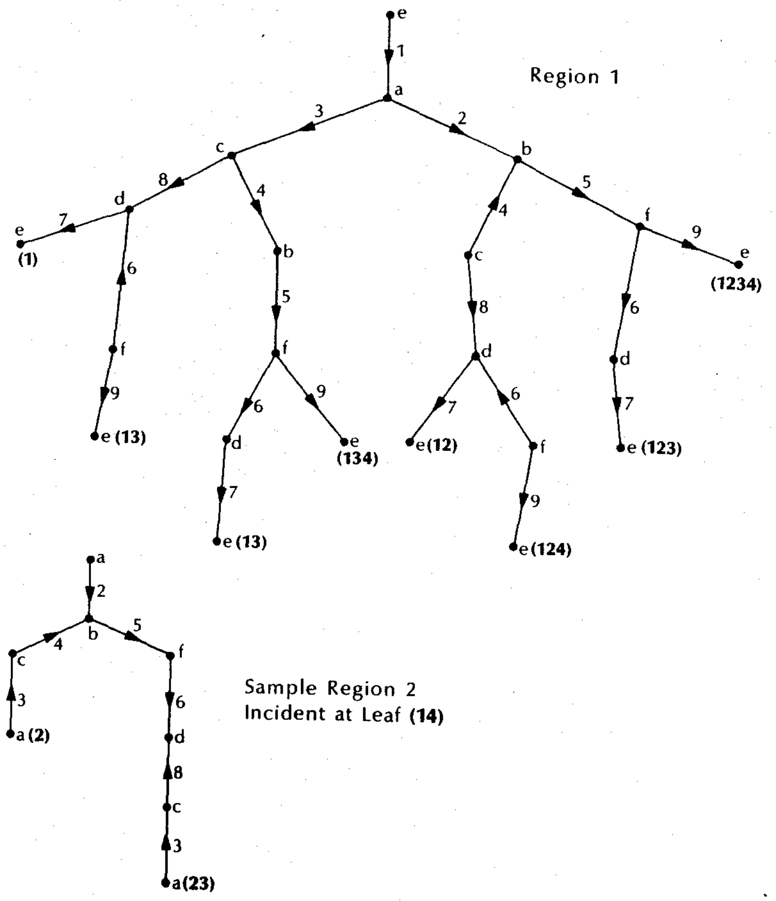


Figure 2. Combinatorial Tree Based on Figure 1 Boundary Network.



root edge, and the set of all circuits is the set of all possible second regions. This is illustrated in Figure 2 in the case of leaf (14). Since (23) leaves no remaining zones for a third region, the only option for three regions which includes (14) is (14, 2, 3). Completing the tree shows that $P(4,3) = 5$.

In enumerating solutions for m regions the algorithm begins by turning left at each junction, producing the right-most leaf in Figure 2. Upper bounds can be imposed at each vertex in the tree and used to bound the search, but lower bounds can be imposed only at each leaf. Each feasible leaf is immediately used to develop the tree for the second and subsequent regions: when these have been searched the algorithm backtracks into the first region's tree. In all cases backtracking is to the first encountered left turn; the search then proceeds to turn right at that vertex, until all of the first region tree has been traversed.

The next section describes the data structure used, and this is followed by a discussion of computational experience.

DATA STRUCTURE

The algorithm clearly requires more than a simple adjacency matrix; the data structure used is a simplification of one used to represent boundary networks for cartographic purposes.

For the purpose of the data structure each edge has a direction associated with it. This direction is arbitrary, and results in our case from the process of digitizing. Each edge is represented by a record, containing eight items, as follows:

1. Zone to the right of the edge.
2. Zone to the left.
3. Edge reached by traversing in forward direction and turning right.
4. As 3, turning left.
5. Edge reached by traversing in backward direction and turning right.
6. As 5, turning left.
7. Vertex at beginning of edge.
8. Vertex at end of edge.

Vertices, edges and zones are assumed to be numbered with consecutive, positive integers. For the edge pointers, items 3 through 6 above, the edge number is made positive if the adjacent edge points in the same direction and negative if it points in the opposite direction. Table I shows the data structure applied to the network in Figure 1.

TABLE I

Data structure for Figure 1

Record	Right Zone	Left Zone	Top Right Edge	Top Left Edge	Bottom Right Edge	Bottom Left Edge	From Vertex	To Vertex
1	1	0	3	2	9	7	e	a
2	2	0	-4	5	1	-3	a	b
3	1	2	8	4	-2	1	a	c
4	3	2	5	-2	3	-8	c	b
5	3	0	6	9	2	4	b	f
6	3	4	-8	7	-9	5	f	d
7	1	4	1	-9	6	8	d	e
8	1	3	7	-6	-4	3	c	d
9	4	0	-7	1	5	-6	f	e

In our implementation the data structure is produced by an interactive digitizing routine. The operator digitizes each line in the boundary network once, and the routine searches for intersections, forms vertices and breaks lines into edges as the digitizing proceeds. A pre-processor is then used to adjust the network so that all vertices are 3-valent. The coordinate data produced by the digitizing is in a second random access file pointed to by the file of edge records, and is not used by the region-building algorithm or the pre-processor.

COMPUTATIONAL EXPERIENCE

The algorithm has been coded in Fortran for the Digital DEC-10 KL10 processor. It is currently dimensioned to 150 zones and 400 edges. Tests were made on the 21 planning districts of London, Ontario and are reported in Table II. Populations were based on the 1971 Census of Canada. In those cases where bounds are shown as optimized, the initial upper and lower bounds were replaced by the maximum and minimum region populations of the first solution. Successive replacement gave a single solution with minimum population range.

Table II

Computational experience for $n = 21$

Regions	Population bounds Lower	Upper	Number of solutions	cpu time (seconds)
3	70,000	80,000	72	59
3	71,500	75,500	8	17
3	72,500	74,500	3	10
2	110,257*	110,322*	1	50
4	53,554*	57,472*	1	11

Total population: 220,579

*Optimized bounds

The computing times compare favourably with those given by Rossiter and Johnston. They appear comparable to those reported by Garfinckel and Nemhauser, although additional constraints in the form of the exclusion rules and bounds on shape used in their algorithm would presumably improve performance considerably. However, it is likely that when constraints are sufficiently tight that the number of feasible regions is relatively small, the Garfinckel and Nemhauser two-stage algorithm may well be more efficient for some applications. Both methods are capable of enumerating all feasible solutions: their relative efficiencies must depend on the specific details of a given problem.

CONCLUSIONS

The algorithm described in this paper is based on a representation of the region-building or polygon aggregation problem as a combinatorial tree. It requires a data structure which is considerably richer than the simple adjacency matrix, and uses a set of simple rules to transform a boundary network into a tree given an arbitrarily chosen root edge and root vertex.

The efficiency of the algorithm depends on the bounds and constraints applied to limit the size of the combinatorial tree. In the electoral districting problem where the constraints are likely to produce a very small set of feasible solutions, our work supports the conclusion of Garfinckel and Nemhauser that $n = 40$ represents a rough upper limit to the size of problem which can be handled by branch and bound methods in reasonable cpu time.

REFERENCES

1. Cliff, A.D. and P. Haggett, "On the efficiency of alternative aggregations in region-building problems," Environment and Planning, 2, 1970, 285-294.

2. Garfinckel, R.S. and G.L. Nemhauser, "Optimal political districting by implicit enumeration techniques," Management Science, 16, 1970, B495-B508.

3. Rossiter, D.J. and R.J. Johnston, "Program GROUP: the identification of all possible solutions to a constituency-delimitation problem," Environment and Planning A, 13, 1981, 231-238.

4. Minciardi, R., P.P. Puliafito and R. Zoppoli, "A districting procedure for social organizations," European Journal of Operational Research, 8, 1981, 47-57.