Mathematical Programming 42 (1988) 159-169
North-Holland

EUROPEAN ASSEMBLY CONSTITUENCIES FOR WALES – COMPARING OF METHODS FOR SOLVING A POLITICAL DISTRICTING PROBLEM

Bjørn NYGREEN

Division of Economics The Norwegian Institute of Technology, Trondheim, Norway

Received January 1988

This paper considers the problem of grouping Parliamentary constituencies for Wales together into European constituencies in such a way that the resulting European constituencies have as equal electorates as possible. Three different solution methods are compared. These methods are integer programming, set partitioning and implicit enumeration. Computational results obtained by use of SCICONIC/VM and user written programs on a DEC VAX 8600 are reported.

1. Acknowledgement

This work was started during my sabbatical stay with Scion Ltd. in 1984/85. I am grateful to Scicon's Management Science Division for giving me an office and computer resources to carry out parts of this work.

While I stayed with Scicon, the late Professor E.M.L. Beale, FRS, received a request from "The Office of Population Censuses and Surveys" in London about methods for grouping Parliamentary constituencies together into European constituencies. This request initiated this work, and I am grateful to Martin Beale for encouraging me while I worked on this problem in Milton Keynes.

In the early stages of the work, I had very useful discussions with Martin Beale, and the concept of rooted trees used in this paper was suggested to me by him.

Together with Martin Beale, I decided to use Wales as an example in my research. We also decided first to try how far it was possible to reach solutions by use of standard software for mathematical programming, which was what I did on this problem while I stayed with Scicon Ltd.

After returning to Trondheim, I have done some more work on the problem. Here I present an integer programming formulation together with two other ways of handling the given geographical clustering problem.

2. Introduction

The problem discussed in this paper is the problem of grouping known Parliamentary constituencies (PCs) into a given number of European constituencies (ECs). The goal for the grouping will be to get connected European constituencies with as equal electorates as possible.

This paper reports results for Wales with 38 PCs and 4 ECs. But the models discussed here may also be used for England.

The boundaries of all the PCs are taken as given. This means that the grouping problem is a pure combinatorial one. The problem is best illustrated by use of a connected graph, as in Figure 1, where all PCs are represented by nodes, and all common boundaries between PCs are represented by arcs. The names of the PCs, numbered in Figure 1, can be found in the solution given in Appendix 1.

The grouping problem can then be stated as: "Delete arcs from the graph until the graph becomes a forest with as many trees as the given number of ECs. Then let each tree represent an EC".

This sort of graph-partitioning is neither mentioned in the standard textbook on optimization and graphs by Minieka (1978), nor in the latest bibliography on combinatorial optimization by O'h Eigeartaigh et al. (1985). But Foulds (1981) presents two pages on an integer programming formulation of a political redistricting problem in his textbook. This formulation however does not say anything about connectedness.

Such political districting problems are, however, covered in several other places in the literature. Hess et al. (1965) described a heuristic method based on warehouse-location which takes both connectedness and compactness into account. Garfinkel and Nemhauser (1970) give a two-step implicit enumeration technique for solving political redistricting problems. Smith, Foulds and Read (1976) give a heuristic

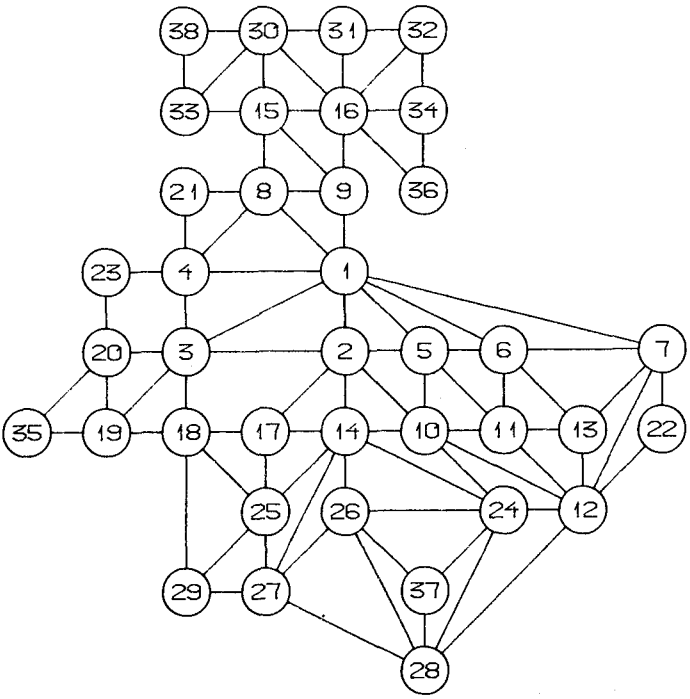

Fig. 1. PC-graph for Wales.

solution procedure for optimization of the compactness under a given maximum electorate deviation.

After doing some work with data for Wales, I got some "banana" shaped ECs when no compactness measure was used. After checking all of today's ECs both for Wales and England, I found that it was possible to represent all these ECs as rooted trees of maximum depth two in the PC-graph in Figure 1. From this I decided to exclude all possible ECs that could not be represented as a maximum depth two trees. No other measure of compactness has been used.

In practice one might also consider some political constraints other than connectedness and compactness. Of such constraints I have chosen to force all PCs in a city into the same EC. One simple way of doing this is simply to merge their nodes in the PC-graph, but this creates difficulties in what maximum depth two trees really means. For this reason I have chosen not to merge the nodes for different PCs in the same city at the problem specification stage.

To get as equal EC electorates as possible, I have chosen to minimise the sum of squares of deviation of EC electorates from the electorate mean. The reason for using squares is mainly to let one large deviation count more than several small deviations. The squares are represented by a piecewise linear approximation between a few given breakpoints.

Both the electorates for each PC and the boundaries between PCs have been taken from the Boundary Commission for Wales (1984). Boundaries that have only one point in common have not been defined as common boundaries.

This paper presents and compares the following solution methods for the stated problem: integer programming, set partitioning and implicit enumeration.

3. An integer programming formulation

The model is written in such a way that duplicate real solutions are possible for PC-graph in Figure 1 as there are ECs. One difficult task of formulating a combinatorial problem as an integer program is to avoid as many duplicate solutions as possible.

In trying to achieve this I have introduced new secondary arcs in the graph. Such arcs are arcs between nodes that have at least one common neighbour without being neighbours themselves. In the new graph, all the ECs will be represented as rooted trees of maximum depth one.

3.1. *Some parts of the model without combinatorial preanalysis*

Some data definitions:

I: number of PCs.
N: number of ECs.
R: maximum relative deviation from the average for an ECs electorate.
S: set of suffices for nodes which will be considered as possible roots in the trees.

Definition of some variables:

δ_i: =1 if PC i is a root in an EC, otherwise =0, $\forall i \in S$.
w_{ji}: =1 if old arc (j, i) is an arc in an EC-tree, otherwise =0, $\forall i \in S$.
u_{ki}: =1 if new arc (k, i) is an arc in an EC-tree, otherwise =0, $\forall i \in S$.

Some of the model-constraints:

The number of roots must be equal to the given number of ECs, this means:

$$\sum_{i \in S} \delta_i = N.$$

All used old arcs go into roots. This means that the following constraints must hold:

$$w_{ji} - \delta_i \leq 0 \quad \forall i \in S \text{ and } (j, i) \text{ is an old arc.}$$

To have a new arc from k into a root i, one also needs an old arc from at least one neighbour of k into i.

$$u_{ki} - \sum_j{}' w_{ji} \leq 0 \quad \forall i \in S \text{ and } (k, i) \text{ is a new arc.}$$

The prime indicates that the summation over j in the last constraint should be done over all j that are common neighbours of k and i. For each node that is not a root, one will either have an old or a new outgoing arc. This means:

$$\delta_i + \sum_j w_{ij} + \sum_k u_{ik} = 1 \quad \forall i.$$

In the introduction I mentioned that all PCs in the same city should belong to the same EC. These constraints are handled implicitly in the definitions of roots and arcs in the matrix generator.

The simplest way to describe the discrete part of the problem is to say that δ_i, w_{ij} and u_{ik} are all binary variables. But Hummeltenberg (1984) as one of many says that one should use special ordered sets, as introduced by Beale and Tomlin (1970) where possible. I will therefore specify that all variables in the last constraint for each i form an S1-set.

In addition to the combinatorial part of the model presented above, the simplest version of the model includes: variables and equality constraints used to calculate the electorate deviations, upper bounds on the deviations, and an objective function that minimises a piecewise linear approximation of the sum of square electorate deviations.

In this first model we do no combinatorial analysis at the matrix generation stage. This means that all nodes go into the set of possible roots.

3.2. Combinatorial preanalysis

The model is written in such a way that duplicate real solutions are possible for different sets of roots, but not after fixing the roots. This means that one should expect the overall computer time to depend heavily on the number of possible roots. Some sort of combinatorial analysis to restrict the number of nodes in S should therefore be useful.

Root rule

(i) Initially set $S = \{1, 2, \ldots, I\}$.

(ii) For $i = I, I-1, \ldots, 2, 1$. Loop over each node and do (iii).

(iii) Generate all valid ECs with node i as the root, and do (iv) for each generated EC.

(iv) For each EC, try to use another node $j \in S$ as a root in a maximum depth two tree. If such a $j \neq i$ can be found for all ECs with i as the root, then remove node i from S.

For each EC generated in part (iii) it is possible to look at the graph that remains when all nodes in the EC are removed from the PC-graph. If there is a component in this remainder which cannot be an integer number of ECs then the generated EC is not valid.

This rule will in most cases work better if the nodes are numbered in such a way that nodes in the outer part of the graph are given high numbers. This is to try to remove the nodes in the outer part of the graph first.

I found it useful to introduce some sort of cut constraints explicitly saying that it should be possible to reach at least one possible root from each node. One should compare these constraints and drop all cuts that are dominanted by others. If a non-dominant cut involves only one node, then this node must be a root.

4. A set partitioning approach

Garfinkel and Nemhauser (1970) use a two stage approach, where possible districts are generated in stage one and set together in stage two.

A similar approach is used here.

4.1. Generation of valid ECs

Some or all valid ECs may be generated in different ways, I have chosen to do it in the following way:

(i) First reduce the number of roots to consider by use of the root rule in 3.2.

(ii) For each root in S, use the graph structure to generate and store all valid ECs, using the "graph that remains" test mentioned in 3.2.

(iii) Do some sort of a test to delete duplicate ECs generated for different roots.

(iv) Calculate the square percentage electorate deviations for all non-deleted ECs.

In part (ii) one has to check that if one PC from a city is in the EC, then all the PCs from the city must be in the EC.

4.2. Set partitioning

The problem of selecting some of the generated ECs so that each PC belongs to one and only one of the selected ECs can easily be formulated as a set partitioning problem. See Balas and Padberg (1976) for a survey of set partitioning problems.

I found it useful to add an extra constraint to the set partitioning problem saying that exactly N PCs should be selected.

In the mentioned work of Garfinkel and Nemhauser, they use some sort of an implicit enumeration technique to find the best of the generated districts. In this paper the set partitioning problem is solved by use of standard MP-software.

5. An implicit enumeration approach

A sketch of an implicit enumeration approach is given here.

5.1. Root list generation

To create a sorted list of possible roots that can be used in the enumeration, one may do as described here:

(i) First use the root rule in 3.2 to find S.

(ii) Generate the cuts in 3.2.

(iii) If any roots have been fixed, then put them into the rootlist from the beginning. Sort them such that the root with most other nodes fixed to it comes first.

(iv) For all non-fixed nodes, $i \in S$, set $P_i = I$.

(v) For all non-fixed nodes, $i \in S$, that are in at least one generated cut, recalculate P_i as

$$P_i = \min\{\text{number of } \delta_j\text{'s in a cut containing } \delta_i\}$$

(vi) Put j into the rootlist before k if $P_j < P_k$. Break ties first for the node that is in the highest number of cuts, then for the smallest node number.

5.2. A sketch of the implicit enumeration technique used

The implicit enumeration is done by using some sort of a "tree search". A general description of this type of algorithm is given by Geoffrion (1967). The enumeration starts by picking a root from the beginning of the root list.

Here we use the fact that all valid ECs are rooted trees of maximum depth two. The "graph that remains" test mentioned in 3.2 is also used here. Finally we use the fact that while we have some current ECs with their square percentage electorate deviation, the best we can achieve in the search from here is to divide the rest of the electorate equally among the rest of the ECs. This will be used as a backtracking bound.

The constraint saying that all PCs in a city shall go into the same EC can easily be taken care of in the enumeration.

6. Computational results

All the reported results are found by use of a DEC VAX 8600 under VAX/VMS V4.4. All the used software is written in Fortran, either by me or by Scicon Ltd.

The optimal solution is given in Appendix 1. The sum of square percentage deviation for this solution is 2.36.

One must expect the running times to depend heavily on the maximum allowed relative deviation for an EC-electorate. The recommendation from the Boundary Commission (1984) has a maximum relative deviation of 6.6%. From this I decided to try $R = 5\%$ when I first solved the problem. The optimal solution has a maximum relative deviation of 1.3%. From this I have chosen to present results for $R = 1.5\%$ (a bit more than 1.3%) and for $R = 4.5\%$ (3 times 1.5%). The reason for having the second R as an integer multiple of the first R, is to make it easy to get equal linear approximations in the integer programming models.

6.1. Integer programming

The main integer programming model is described in 3.1-3.2. For this model results are given for both R-values. To show how important the combinatorial analysis can be, I also report results for the model in 3.1 for $R = 1.5\%$.

The matrix generation system MGG/VM version 2.11 from Scicon (1985) has been used to generate all the matrices used.

The sizes of the generated integer programs are given in Table 1 together with three depths and node numbers from the branch and bound search.

The tree depth is the depth in the branching-tree where the integer optimum was found. The number of nodes is the node number in which the optimal solution was found and the total number of nodes needed to complete the search.

SCICONIC/VM version 1.42 from Scicon (1986) has been used to solve all the integer programs. The reported solutions have been found by use of the agenda PRIMAL and GLOBAL (PRESOLVE).

Table 1

Matrix sizes and branching information

Maximum relative deviation	1.5%	4.5%	1.5%
Combinatorial analysis	yes	yes	no
Rows	176	262	615
Columns	202	435	676
Number of $S1$-sets	22	25	38
Number of set members	144	214	486
Number of binary variables	10	16	38
Tree depth for integer optimum	6	11	16
Nodes for integer optimum	162	40	1017
Total number of nodes	165	457	1363

Table 2
Set partitioning matrices

Maximum relative deviation	1.5%	4.5%	1.5%
Combinatorial analysis	yes	yes	no
Rows	40	40	40
Columns	41	564	132

Table 3
CPU-seconds

Maximum relative deviation	1.5%	4.5%	1.5%
Combinatorial analysis	yes	yes	no
Integer programming	73	413	4976
Set partitioning	18	47	20
Implicit enumeration	15	43	87

6.2. Set partitioning

The matrix generator for this problem is user written in Fortran. The resulting set partitioning problem with one extra constraint is solved by use of the agendum PRIMAL in SCICONIC/VM version 1.42.

The results for matrix sizes are given in Table 2.

6.3. Implicit enumeration

The computer program for this solution method is in user written Fortran.

6.4. Used computer time

The CPU-seconds reported in Table 3 include time for combinatorial preanalysis, matrix generation and optimization.

7. Conclusions

The computational results show that implcit enumeration and set partitioning work better than integer programming. The tables also show that there is computer time to be saved by guessing the maximal relative deviation in the optimal solution.

The integer programming results in Table 3 show that one has to use combinatorial analysis if one wants to try integer programming. If one compare $R = 1.5\%$ and $R = 4.5\%$ in Table 1, one sees that the estimation has worked much better for

$R = 4.5\%$ than for $R = 1.5\%$ from the fact that the integer optimum was found for a much smaller node number for the largest R. From the tree depths for the integer optima one sees that a lucky estimation would have found the optimal solution very quickly.

The optimal solution presented in Appendix 1 shows a sum of square percentage deviations of 2.36 compared with 68.18 for the solution recommended by the Boundary Commission (1984).

Table 4

EC-no.	PC-no.	PC-name	PC-elect.	EC-elect. & rel. dev.
1	1	Brecon and Radnor	47853	531608
	2	Cynon Valley	50856	&
	6	Blaenau Gwent	56301	−0.62%
	7	Monmouth	56961	
	9	Montgomery	37928	
	11	Islwyn	50806	
	12	Newport West	54900	
	13	Torfaen	59555	
	17	Rhondda	63183	
	22	Newport East	53265	
2	5	Methyr Tydfil and Rhymney	60210	532283
	10	Caerphilly	64173	&
	14	Pontypridd	61624	−0.49%
	24	Cardiff North	54012	
	26	Cardiff West	59223	
	27	Vale of Glamorgan	63728	
	28	Cardiff South and Penarth	60231	
	29	Bridgend	54673	
	37	Cardiff Central	54409	
3	3	Neath	55855	533850
	4	Carmarthen	64297	&
	18	Aberavon	54059	−0.20%
	19	Swansea East	58010	
	20	Gower	57336	
	21	Pembroke	68815	
	23	Llanelli	64562	
	25	Ogmore	52053	
	35	Swansea West	58863	
4	8	Ceredigion and Pembroke North	61155	541873
	15	Meirionnydd Nant Conwy	30798	&
	16	Clwyd South West	56520	1.30%
	30	Conwy	52197	
	31	Clwyd North West	63201	
	32	Delyn	63300	
	33	Caernarfon	44795	
	34	Alyn and Deeside	57432	
	36	Wrexham	61472	
	38	Ynes Mon	51003	

From this I am convinced that it is possible to find a solution with a much more equally distributed electorate than the recommendation from the Boundary Commission, even if one imposes new political constraints.

If one should try to solve a similar problem for England, I would recommend trying the set partitioning method first. The reason for this is that I would expect the running time for the implicit enumeration to grow faster with the problem size than the running time for the EC-generation part of the set partitioning problem. This is also seen from comparing the cases without the root rule in Table 3.

A special purpose set partitioning software will probably solve the generated set partitioning problem faster than SCICONIC.

For England with more than 500 PCs and more than 60 ECs, the problem may be too large to solve in reasonable computer time.

In practice this will not be any real problem, because I would expect to get a real reduction in the sum of squared electoral deviations, even if one divides England into regions containing from 5 to 20 of todays ECs and solves separate problems for each region.

One would also have to model the political constraints in another way, because one cannot put all of London's PCs into one EC.

Appendix 1. Optimal European constituencies for Wales

The optimal solution for the problem discussed in this paper is listed in Table 4. The sum of square percentage deviations for the ECs is *2.36*. The ECs are numbered such that the Ecs electorate increases with increasing EC-number. The PC-numbers are the numbers used in 3.2 and in Figure 1.

References

E. Balas and M.W. Padberg, "Set partitioning: A Survey," *SIAM Review* 18 (1976) 710–780.

E.M.L. Beale and J.A. Tomlin, "Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables," in: J. Lawrence, ed., *Proceedings of the Fifth International Conference on Operational Research* (Tavistock Publications, London, 1970) pp. 447–454.

Boundary Commission for Wales, *1983 Review of European Assembly Constituencies* (Her Majesty's Stationary Office, London, March 1984).

L.R. Foulds, *Optimization Techniques, An Introduction* (Springer-Verlag, New York, 1981).

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

A.M. Geoffrion, "Integer programming by implicit enumeration and Balas' method," *SIAM Review* 9 (1967) 178–190.

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

W. Hummeltenberg, "Implementation of special ordered sets in MP software," *European Journal of Operations Research* 17 (1984) 1–15.

E. Minieka, *Optimization Algorithms for Networks and Graphs* (Marcel Dekker, Inc., New York, 1978).
M. O'h Eigeartaigh, J.K. Lenstra and A.H.G. Rinnooy Kan *Combinatorial Optimization, Annotated Bibliographies* (John Wiley & Sons, New York, 1985).
Scicon, *MGG/ VM User Guide* (Scicon Ltd, Milton Keynes, August 1985).
Scicon, *SCICONIC/VM User Guide* (Scicon Ltd, Milton Keynes, February 1986).
R.G. Smith, L.R. Foulds and E.G. Read, "A political restricting problem," *New Zealand Operational Research* 4 (1976) 37-52.