Mathematical Programming 3 (1972) 339–344. North-Holland Publishing Company

AN INTEGER PROGRAMMING APPROACH TO A CLASS OF COMBINATORIAL PROBLEMS

E.M.L. BEALE and J.A. TOMLIN*

Scicon Ltd., London, England

Received 20 December 1971
Revised manuscript received 25 September 1972

In this paper, we discuss the solution of a class of modified quadratic assignment problems, with particular reference to an application involving decentralization of a large organization. The main emphasis is on the use of a standard branch and bound mathematical programming system (UMPIRE) and the problem manipulations required to carry this out efficiently.

1. Introduction

In this paper, we discuss our experience with the solution of a location problem which can be formulated as a slightly modified quadratic assignment problem [4]. The specific example we consider concerns the relocation of departments of the British Civil Service at present situated in London to other areas in Britain, with the monetary objective of reducing the cost of office accomodation and the social objective of providing employment in development areas. This problem has been considered in some detail in Elton et al. [2], who devised a specially tailored branch and bound algorithm for solving the problem. In contrast, we have chosen to adopt an integer programming approach to demonstrate the feasibility of using a production branch and bound integer programming code (such as Scicon's UMPIRE system) to tackle problems of this type and the wide class of other combinatorial problems which can be formulated in terms of quadratic assignment.

Since the areas of application of the model are quite general, we shall refer simply to facilities being sited at various locations, with a fixed cost (or benefit) associated with each assignment and a "communications cost" between two facilities determined by the amount of communication and the distance between them, or more generally the cost

* Now at Stanford University, Stanford, Calif., U.S.A.

per unit communication between their respective sites. The only feature special to this application are side constraints specifying that the number of persons employed at these facilities at any site must fall between an upper bound and a lower bound if any facility is located at this site. Some such restriction is clearly necessary if the social objective is to be attained, otherwise all the facilities would simply be located at the cheapest site.

2. Integer programming formulation

We now specify the integer programming model of the facilities location problem in much the same form we use for Scicon's MGG matrix generator package.

Indices:

$1 \leqslant i < k \leqslant m$ (facility indices),
$1 \leqslant j, l \leqslant n$ (location indices).

Constants:

m = number of facilities;
n = number of locations (for our problems $n < m$);
C_{ik} = amount of communication between facilities i and k;
D_{jl} = cost per unit of communication between sites j and l;
G_{ij} = cost (or benefit) of locating facility i at site j;
E_i = number of persons employed by facility i;
U_j = upper bound on employment at site j;
L_j = lower bound on employment at site j if any facility is located there;
N_i = number of facilities communicating with facility i (i.e., the number of non-zero C_{ik}, $k = 1, \ldots, m$).

Variables:

δ_{ij} = $\begin{cases} 1 & \text{if facility } i \text{ is located at } j, \\ 0 & \text{otherwise;} \end{cases}$

x_{ijkl} = $\begin{cases} 1 & \text{if } i < k,\ C_{ik} \neq 0 \text{ and } \delta_{ij}\,\delta_{kl} = 1, \\ 0 & \text{otherwise;} \end{cases}$

ξ_j = $\begin{cases} 1 & \text{if any facility is located at } j, \\ 0 & \text{otherwise;} \end{cases}$

s_j = non-negative slack on employment upper bound at j.

Note that x_{ijkl} is defined only for $i < k$ and $C_{ik} \neq 0$ since we wish to consider only connected pairs of facilities and each of these pairs only once. The problem is then:

Objective:

Minimize $$\sum_{i,j} G_{ij}\,\delta_{ij} + \sum_{i<k}\sum_{j,l} C_{ik}\,D_{jl}\,x_{ijkl}. \tag{2.1}$$

Constraints:

$$\sum_{k>i}\sum_{l} x_{ijkl} + \sum_{k<i}\sum_{l} x_{klij} - N_i\,\delta_{ij} = 0, \quad \begin{matrix} i = 1,\ldots,m, \\ j = 1,\ldots,n, \end{matrix} \tag{2.2}$$

$$\sum_{i} E_i\,\delta_{ij} - U_j\,\xi_j + s_j = 0, \quad j = 1,\ldots,n, \tag{2.3}$$

$$(L_j - U_j)\,\xi_j + s_j \leqslant 0, \quad j = 1,\ldots,n, \tag{2.4}$$

$$\sum_{j,l} x_{ijkl} = 1, \quad 1 \leq i < k \leq m\,, \tag{2.5}$$

$$\sum_{j} \delta_{ij} = 1, \quad i = 1,\ldots,m. \tag{2.6}$$

The constraints (2.2) in conjunction with (2.5) can be easily shown to imply that the x_{ijkl} must take on the required values given integer δ_{ij}. The constraints (2.3) and (2.4) specify the upper and lower bounds on employment at each site (if any). Finally, (2.6) specifies that the δ_{ij} variables are grouped in multiple choice sets or "special ordered sets of type 1" (see [1]) which can be of great help in reducing the amount of work done in a branch and bound algorithm [1; 5].

This formulation has $mn + 2n + \frac{1}{2}m(m-1) + m$ constraints and a total number of variables depending on the density of the communication matrix C. The last $\frac{1}{2}m\,(m+1)$ constraints are however of the generalized upper bound (GUB) type and hence can be treated implicitly by UMPIRE.

3. Initial computational experience

Our initial test data supplied by the Civil Service Department involved citing 25 facilities at 3 possible locations with quite loose employment constraints (2.3) and (2.4). We began our experiments using a

straightforward last-in-first-out branch and bound approach with penalty calculations to choose the branching variables [5]. The results were very poor. Because of the large number of zero penalties (a chronic occurrence with combinatorial problems), the tree search was almost arbitrary, resulting in an enormous number of branches which produced several rather poor integer solutions, but failed to terminate in 50 min on the Univac 1108.

This performance was improved using the UMPIRE facility to supply a priority ordering of the variables. This is easy to do since it is clear that the location of those facilities with large amounts of communication tends to determine the location of the other facilities. We were thus able to modify the matrix generator to produce a priority list of the special ordered sets (2.6) in order of decreasing value of $\Sigma_k C_{ik}$. Use of the priority list enabled us to obtain the optimum solution and complete the search in 30 min. This is still far from satisfactory for such a small problem, despite the enormous number of possible integer solutions.

The comparative success of the priority ordering does, however, provide the clue to a reformulation of the problem in conjunction with the priorities.

4. Reformulation

The costs in this problem can be classified as direct or first-order costs G_{ij} and indirect or second-order costs $C_{ik}D_{jl}$. If all the costs were first-order, then apart from the side constraints (2.3) and (2.4) we would have a simple assignment problem. It would then be very easy to obtain the best integer solution. With large second order costs, we find it far more difficult to obtain such a solution. However, if as we proceed with the tree search some facilities have already been assigned, say the subset A, then the second-order costs $C_{il}D_{jl}$ can in principle be treated as first-order costs for $i \notin A$, $k \in A$ (where k is assigned to l) and may be added to G_{ij}.

An alternative to our first formulation which does this implicitly can be obtained by replacing the mn constraints (2.2) by

$$\sum_l x_{ijkl} - \delta_{ij} = 0, \tag{4.1}$$

for all j, and i,k such that $i < k$ and $C_{ik} \neq 0$. The difficulty here of course

is the very large number of constraints involved. It is here that the priority ordering comes into play, for by virtue of this ordering, we know in advance the order in which facilities will be located.

We write a constraint of type (4.1) only for those i,k such that k is of higher priority than i, for $C_{ik} \neq 0$, $i < k$. Now let us define S_i as the set of all k with lower priority than i for which $C_{ik} \neq 0$ and let M_i be the cardinality of S_i. Then we write

$$\sum_{\substack{k>i \\ k\in S_i}} \sum_{l} x_{ijkl} + \sum_{\substack{k<i \\ k\in S_i}} \sum_{l} x_{klij} - M_i \delta_{ij} = 0 \qquad (4.2)$$

for each i,j such that i communicates with some facility of lower priority. We now replace (2.2) by the two sets of constraints (4.1) and (4.2). Note that this results in just over half the number of non-trivial constraints which would be obtained using constraints of type (4.1), but considerably more than the original formulation.

The modification of the matrix generator turned out to be a relatively simple matter and the problem was re-run using the "best projection" (BP) criterion type tree search due to J.P.H. Hirst (see [3]), but restricting the search to the two most recently created branches whenever possible. The problem was solved and the search completed in only 5 min total running time – one tenth of the time expended in our first non-optimal effort.

A second more realistic problem, again with 25 facilities but with 6 locations, was then solved. Since doubling the number of locations doubles the number of rows and integer variables and very considerably increases the number of x_{ijkl} variables, we might expect the solution time to increase dramatically. In fact, we were able to obtain guaranteed optima in just under 20 min for a range of values of G_{ij} and D_{kl}. Only when the second-order costs were made very large in comparison to the first-order costs did the problem fail to terminate in 20 min and even then it was clear that no significantly better solution could be found. Complete statistics for this run and one of the terminating runs are given among the examples in [3].

5. Conclusion

It has been recognized for some time that purely combinatorial problems are often very difficult to solve by general integer programming

methods and that it is important to linearly constrain the integer variables as tightly as possible. The need to avoid an excessive number of constraints is however also crucial and a compromise may be difficult. In this instance, we were able to achieve such a compromise by taking advantage in the formulation of the branch and bound strategy to be used in actually solving the problem, a principle we expect will prove to be valuable in general.

References

[1] E.M.L. Beale and J.A. Tomlin, "Special facilities in a general mathematical programming system for non-convex problems using special ordered sets of variables", in: *Proceedings of the 5th IFORS Conference*, Ed. J. Lawrence (Tavistock, London, 1970) pp. 447–454.

[2] M.C.J. Elton, P.M. Hollis, J.M.H. Hunter, D.W. Millon and J.P. Turner, "An approach to the location of government", presented at the *TIMS International Conference, London, 1970.*

[3] J.J.H. Forrest, J.P.H. Hirst and J.A. Tomlin, "Practical solution of large and complex integer programming problems with UMPIRE", *Management Science* (1972), to appear.

[4] T.C. Koopmans and M.J. Beckmann, "Assignment problems and the location of economic activities", *Econometrica* 25 (1957) 57–76.

[5] J.A. Tomlin, "Branch and bound methods for integer and non-convex programming", in: *Integer and non-linear programming,* Ed. J.Abadie (North-Holland, Amsterdam, 1970) pp. 437–450.