Comp. & Maths. with Appls. Vol. 7, pp. 187–194
© Pergamon Press Ltd., 1981. Printed in Great Britain
0097-4943/81/0401-0187/$02.00/0

THE MULTICOMMODITY ASSIGNMENT PROBLEM: A NETWORK AGGREGATION HEURISTIC

JAMES R. EVANS

Department of Quantitative Analysis, College of Business Administration, University of Cincinnati, Cincinnati, OH 45221, U.S.A.

Communicated by E. Y. Rodin

(*Received October* 1978)

Abstract—We present a network-based heuristic procedure for solving a class of large non-unimodular assignment-type problems. The procedure is developed from certain results concerning multi-commodity network flows and concepts of node-aggregation in networks. Computational experience indicates that problems with over fifteen thousand integer variables can be solved in well under ten seconds using state-of-the-art network optimization software.

INTRODUCTION

In this paper we propose an efficient heuristic procedure for solving a class of large 0–1 integer programming problems. The problem considered here is a generalized version of the assignment problem which may also be viewed as a specialization of the multicommodity transportation problem. Hence, the problem is termed the *multicommodity assignment problem.* We also note that the multicommodity assignment problem is distinct from the multidimensional assignment problem that has been addressed in the literature (Pierskalla[1]). The later may be thought of as finding a matching on a three-dimensional lattice of points whereas the former seeeks a covering of an $n \times n$ array with n disjoint two-dimensional matchings (assignments). The heuristic procedure that we shall present draws upon certain theoretical results concerning integrality in multicommodity networks[2, 4] and concepts of node aggregation in networks[7, 11]. Using state-of-the-art primal simplex network codes[13, 15, 16] to implement the procedure, problems with thousands of variables can be solved in seconds in modern computers.

PROBLEM FORMULATION

The multicommodity assignment problem can be stated as determining an optimal assignment for n objects of n different types among two families of sets $\mathcal{S} = \{S_i\}$ and $\mathcal{T} = \{T_i\}$ with $|\mathcal{S}| = |\mathcal{T}| = n$ so that each set contains exactly one object of each type. A typical application is the situation of market testing n products in n different locations over n time periods. Each product must be tested in every location over n time periods. Each product must be tested in every location, but only one product may be tested in any location during any given time period. Another example involves the assignment of management trainees to various positions over time within a company. The reader may note the obvious equivalence of solutions to the multicommodity assignment problem with the structure of Latin squares. Thus, the problem can be viewed as one of findings an optimal Latin square. In this context, an application might be to minimize the amount of time or cost of data collection in constructing a Latin square design for an experiment.

Mathematically, the problem can be formulated as follows: Let $X_{ij}^k = 1$ if an object of type k is assigned to sets S_i and T_j, and 0 otherwise. We then have:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} c_{ij}^k X_{ij}^k \tag{1}$$

$$\sum_{i=1}^{n} X_{ij}^k = 1; \forall j, k \tag{2}$$

$$\sum_{j=1}^{n} X_{ij}^k = 1; \forall i, k \tag{3}$$

$$\sum_{k=1}^{n} X_{ij}^k \leq 1; \forall i, j \tag{4}$$

$$X_{ij}^k = 0, 1; \forall i, j, k. \tag{5}$$

We remark that (4) will always be satisfied as a strict equality, but the inequality will be useful in this presentation. As it stands, the problem is an integer program with n^3 binary variables and $3n^2-n$ independent constraints.

COMBINATORIAL PROPERTIES

Let $\mathcal{P}$ be the convex polytope defined by (2)–(4) and by

$$X_{ij}^k \geq 0; \forall i, j, k \tag{5'}$$

obtained by relaxing the integrality requirements (5). Implicit in this set of constraints is the fact that $X_{ij}^k \leq 1$ for all i, j, k. Let $V(\mathcal{P})$ be the set of extreme points (vertices) of P. Then $V(\mathcal{P})$ can be partitioned into two sets $I(\mathcal{P})$ and $N(\mathcal{P})$ where $I(\mathcal{P})$ is the set of vertices having all integer (0, 1) coordinates and $N(\mathcal{P})$ the remaining set. It is easy to see that any solution to (2)–(5) must belong to $I(\mathcal{P})$; that is, the linear form (1) is minimized at a vertex of $\mathcal{P}$. Except for a special case, however, $N(\mathcal{P})$ is nonempty, thus precluding solutions by linear programming.

The polytope $\mathcal{P}$ is a special case of the following:

$$\sum_{i=1}^{m} X_{ij}^k = a_i^k; \forall i, k \tag{6}$$

$$\sum_{j=1}^{n} X_{ij}^k = b_j^k; \forall j, k \tag{7}$$

$$\sum_{k=1}^{r} X_{ij}^k \leq u_{ij}; \forall i, j \tag{8}$$

$$X_{ij}^k \geq 0; \forall i, j, k \tag{9}$$

where it is assumed that $\sum_i a_i^k = \sum_j b_j^k$ for each k. These constraints describe a class of network flow problems known as multicommodity transportation (problems[2], which are simply generalizations of the ordinary transportation problem of linear programming[3]. Indeed, if $r = 1$ we have an ordinary capacitated problem. In this case, X_{ij}^k represents the flow from source i to sink j of commodity k over an arc having capacity u_{ij}. The parameters a_i^k and b_j^k are, respectively, supplies and demands. The following theorem follows from the constructive results in[2] and [4] and is also proven algebraically in[5]:

THEOREM

The constraint matrix to (6)–(8) is unimodular† if and only if $m \leq 2$ or $n \leq 2$ for all $r \geq 2$.

From Veinott and Dantzig[6], if a_i^k, b_j^k, and u_{ij} are integers, then all extreme points of the corresponding polytope are integral and only if $m \leq 2$ or $n \leq 2$. An immediate consequence is

COROLLARY

If $n \leq 2$, $N(\mathcal{P}) = \emptyset$.

An example of a non-integer extreme point of $\mathcal{P}$ for $n = 3$ is

†Unimodularity implies that every basis has determinant +1 or −1.

$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ 1 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

The multicommodity assignment problem for $n > 2$ is, then, a difficult integer programming problem.

A NODE-AGGREGATION HUERISTIC

Being a pure integer program, the multicommodity assignment problem is NP-complete, and therefore apparently no efficient solution procedures for obtaining optimal solutions exist. For practical problems of any reasonable size one must resort to heuristic methods for obtaining a, hopefully, good solution. The algorithm we shall present is based upon solving a sequence of $n-1$ simple network flow problems.

As noted previously, the multicommodity assignment problem is a special case for the multicommodity transportation problem with n origins, n destinations, and n commodities (Fig. 1). Each commodity has a supply (demand) of one at each origin (destination). In addition, the capacity of each arc (represented by equation (4)) is unity. For this interpretation we choose the inequality form of constraint (4).

By a *node-aggregation*, we mean a grouping of several nodes in a network into a single node, along with aggregation of arcs between appropriate nodes in the grouping and those external to it. An illustration of a node-aggregation for a 3×3 transportation network is shown in Fig. 2. Aggregations of nodes in networks have recently been investigated by Geoffrion[11], Zipkin[7, 8], and Evans[9, 10]. In particular, they may be interpreted as row and column aggregation in linear programs[17].

Consider now the sequence of aggregations relative to Fig. 1 and the multicommodity

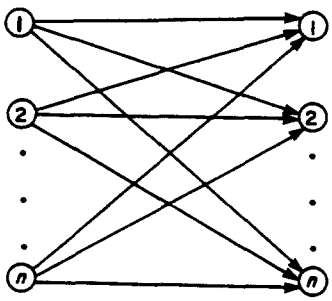

Fig. 1. Network structure of the multicommodity assignment problem.

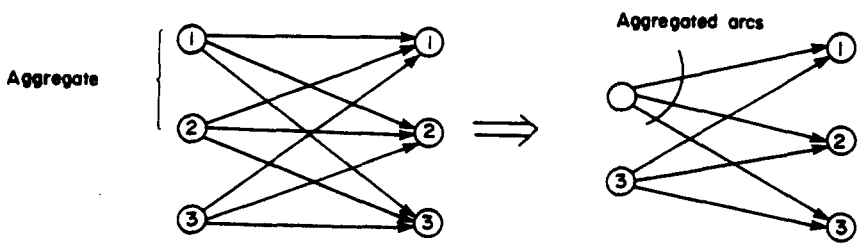

Fig. 2. Node-aggregation.

problem, P_s, $s = 1, 2, \ldots, n-1$, where P_s is defined as follows:

$$P_s\text{: minimize } \sum_{j=1}^{n} c_{sj}^k X_{sj}^k + \sum_{j=1}^{n} d_{s+1,j}^k y_{s+1,j}^k \tag{10}$$

$$\sum_j X_{sj}^k = 1; \forall k \tag{11}$$

$$\sum_j y_{s+1,j}^k = n - s; \forall k \tag{12}$$

$$X_{sj}^k + y_{s+1,j}^k = 1 - \sum_{i=1}^{s-1} \bar{X}_{ij}^k; \forall j,k \tag{13}$$

$$\sum_k X_{sj}^k \leq 1; \forall j \tag{14}$$

$$\sum_k y_{s+1,j}^k \leq n - s; \forall j \tag{15}$$

$$X_{sj}^k, y_{s+1,j}^k \geq 0; \forall j,k. \tag{16}$$

P_s is simply a two-origin multicommodity transportation problem as shown in Fig. 3. This sequence of problems is derived from the original problem as follows. For $s = 1$, we aggregate nodes 2 through n (in Fig. 1) into the single node $s+1$. (For $s = 1$, define $\sum_{i=1}^{0} \bar{X}_{ij}^k = 0, \forall j,k$). The supply at node $s+1$ is simply the sum of the supplies of the aggregated nodes; the flow $y_{s+1,j}^k = \sum_{i=s+1}^{n} X_{ij}^k$; and the capacity of arc $(s+1,j)$ is defined as the sum of the capacities of the aggregated arcs. A surrogate cost $d_{s+1,j}^k$ for the aggregate arcs must be specified. Some appropriate measures are

$$d_{s+1,j}^k = \left(\sum_{i=s+1}^{n} c_{ij}^k \right) / (n - s) \tag{17}$$

$$d_{s+1,j}^k = \min_{s+1 \leq i \leq n} \{c_{ij}^k\} \tag{18}$$

or some other convex combination. In particular, specifying the surrogate cost by (17) results in the concept of *weighted aggregation* (Zipkin[17]). Solving P_1, one obtains a partial solution to the original problem, say $\bar{X}_{1j}^k$, for all j,k. These flows are subtracted from the demands and source 1 is no longer considered. P_2, then, consists of source 2 and the aggregation of sources 3 through n, etc. By successively solving this sequence of problems, a feasible solution to the

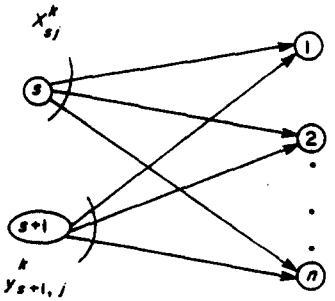

Fig. 3. Aggregated problem P_s.

multicommodity assignment problem is obtained. Feasibility may not always be maintained when solving the general multicommodity transportation problem (with arbitrary integer supplies, demands, and capacities) in this manner. However, when these parameters are unity, the fact that each non-aggregated arc may only carry the flow of one commodity along with the fact that all arcs must be saturated, will guarantee feasibility.

The key to the efficiency of this approach lies in the solution of the aggregated problems P_s, $s = 1, 2, \ldots, n-1$. In[4], it is shown that a $2 \times n$ multicommodity transportation problem with an arbitrary number of commodities can be explicitly transformed into an ordinary, single commodity capacitated transportation problem with n origins and $r+1$ destinations.† When the transformation is applied to the multicommodity assignment problem, arcs incident to the $(r+1)$st destination in the equivalent network will have capacity of zero; thus the aggregated problem reduces to an $n \times n$ capacitated transportation problem. In fact, it can be shown that the number of arcs in the equivalent network with non-zero capacity is exactly $(n-s+1)n$ for P_s. This will be illustrated in the example in the next section. Using the Edmonds-Karp modifications to the out-of-kilter algorithm[12], one can easily demonstrate that the hueristic procedure is polynomially bounded. For computational purposes, however, a primal simplex network code, GNET[13, 14], was used by the author. GNET and other primal codes[15, 16] represent the fastest network optimization codes available to date.

A NUMERICAL EXAMPLE AND COMPUTATIONAL REMARKS

In this section we present a small numerical example to illustrate the procedure and the transformed networks. Table 1 exhibits the c_{ij}^k values for a problem with $n = 4$. A simple average was used (equation (17)) for the surrogate costs for aggregated arcs. Problem P_1 is shown in Fig. 4 and its equivalent transformed network in Fig. 5. In Fig. 5, an arc from node j to node k represents the variable y_{2j}^k. The slack on the arc corresponds to X_{1j}^k. Cost on the arc (j,k) is $(d_{2j}^k - c_{1j}^k)$. The solution to P_1 (non-zero X_{ij}^k's) is:

$$X_{14}^1 = 1$$
$$X_{13}^2 = 1$$
$$X_{11}^3 = 1$$
$$X_{12}^4 = 1.$$

Table 1. Cost coefficients for example problem

i \ j	1	2	3	4
1	10	8	5	10
2	14	6	10	12
3	3	8	14	15
4	10	10	5	10

$k = 1$

i \ j	1	2	3	4
1	4	6	9	12
2	20	30	40	50
3	16	20	45	15
4	20	20	12	12

$k = 2$

i \ j	1	2	3	4
1	10	10	10	10
2	15	20	25	20
3	30	25	20	15
4	10	5	8	14

$k = 3$

i \ j	1	2	3	4
1	16	20	15	18
2	5	30	10	20
3	40	50	12	20
4	16	30	25	25

$k = 4$

†Actually, the result is stated for the $m \times 2$ case, but the problem is symmetric by reversing the role of origins and destinations.

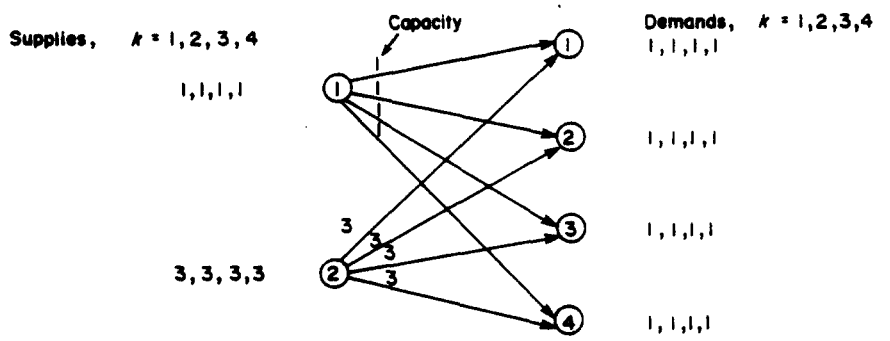

Fig. 4. Problem P_1.

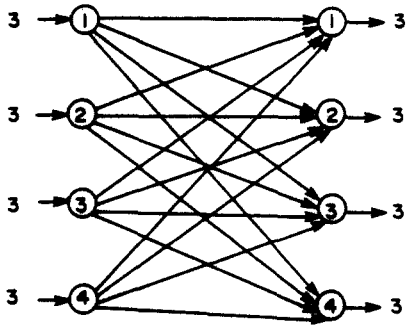

Fig. 5. Transformed network P_1 (all capacities are 1).

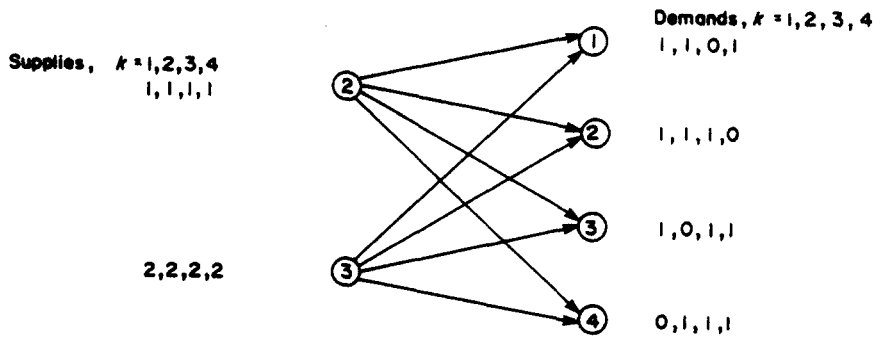

Fig. 6. Problem P_2.

P_2 is given in Fig. 6 and its transformation in Fig. 7. The solution to P_2 is

$$X_{23}^1 = 1$$
$$X_{22}^2 = 1$$
$$X_{24}^3 = 1$$
$$X_{21}^4 = 1.$$

Finally, P_3 and its transformation are given in Figs. 8 and 9 respectively. The remainder of the final solution is

$$X_{32}^1 = X_{41}^1 = 1$$
$$X_{31}^2 = X_{44}^2 = 1$$
$$X_{33}^3 = X_{42}^3 = 1$$
$$X_{34}^4 = X_{43}^4 = 1.$$

Note that the transformed network for P_{s+1} can easily be derived from P_s by deleting (or

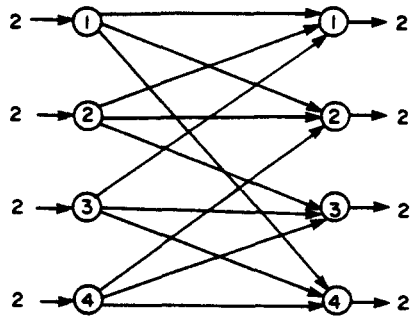

Fig. 7. Transformed network P_2.

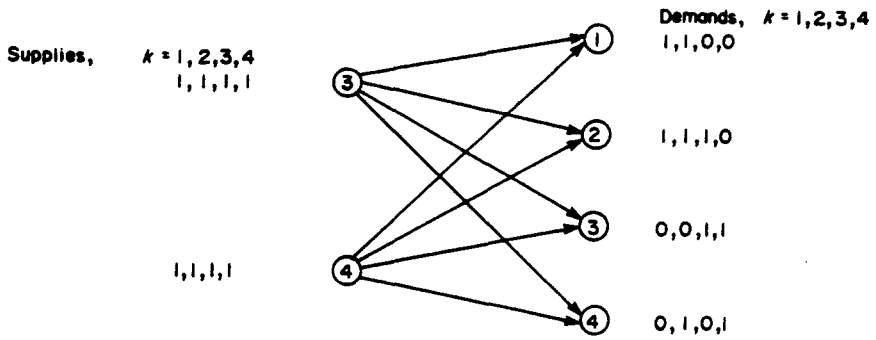

Fig. 8. Problem P_2.

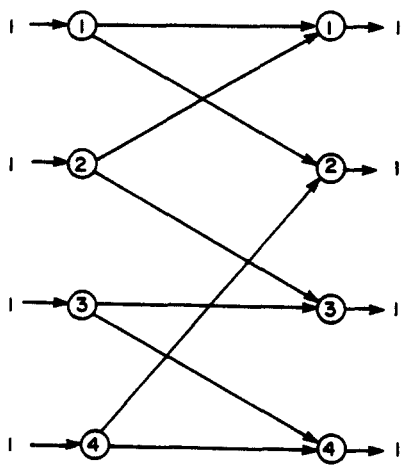

Fig. 9. Transformed network P_3.

effectively setting the capacity to zero) exactly n arcs. Thus the procedure can be efficiently implemented in an interactive mode on a computer with proper updates of supplies, demands, and costs. Using this procedure, problems with very large numbers of variables can be easily solved. For instance, the author has solved problems of the form P_1 for n as large as 25 using GNET[13, 14]. The average time for $n = 25$ was 0.315 sec CPU on an Amdahl 470. Thus a very conservative estimate (since the number of arcs decreases linearly) for solving the sequence of problems $P_1, P_2, \ldots, P_{24}$ would be $(0.315) \times 24 = 7.56$ sec. Considering that fact that for $n = 25$, the problem consists of 15,625 integer variables, this represents a very efficient procedure.

REFERENCES

1. W. P. Pierskalla, The multidimensional assignment problem. *Ops Res.*, **16**, 422–431 (1968).
2. J. R. Evans, J. J. Jarvis and R. A. Duke, Graphic matroids and the multicommodity transportation problem. *Math. Prog.* **13**, 323–328 (1977).

3. G. Hadley, *Linear Programming*. Addison-Wesley, Reading, Mass. (1962).
4. J. R. Evans, A combinatorial equivalence between a class of multicommodity flow problems and the capicitated transportation problem, *Math. Prog.* **10**, 141–144 (1976).
5. K. Truemper, Unimodular matrices of flow problems with GUB constraints. *Networks* 7, 343–358 (1977).
6. A. F. Veinott and G. B. Dantzig, Integral extreme points. *SIAM Rev.* **10**, 371–372 (1968).
7. P. H. Zipkin, *Bounds for aggregating nodes in network problems*. Research Paper No. 79A, Graduate School of Business, Columbia University (1978).
8. P. H. Zipkin, *Error bounds for aggregated convex transportation problems*. Research Paper No. 93A, Graduate School of Business, Columbia University (1978).
9. J. R. Evans, *Integral aggregation in multicommodity networks*. Dept. of Quantitative Analysis, University of Cincinnati (1978).
10. J. R. Evans, Aggregation in the generalized transportation problem, *Comput. Ops Res.* **6**(4), 199–204 (1979).
11. A. Geoffrion, *Customer aggregation in distribution modeling*. Working Paper No. 259, Management Science Study Center, UCLA, (Oct. 1976).
12. J. Edmonds, *An introduction to matching*, Lecture Notes, University of Michigan Summer Conference (1967).
13. G. H. Bradley, G. G. Brown and G. W. Graves, Design and implementation of large scale primal transshipment algorithms. *Mngmt Sci.*, **24**, 1–34 (1977).
14. G. H. Bradley, G. G. Brown and G. W. Graves, *GNET: A primal capacitated network program*. Copyright (1975).
15. F. Glover, D. Karney and D. Klingman, Implementation and computational study on start procedures and bias change criteria for a primal network code. *Networks* 4, 191–212 (1974).
16. V. Srinivasan and G. L. Thompson, Accelerated algorithms for labeling and relabeling of trees with applications to distribution problems. *J.A.C.M.* **19**, 712–726 (1972).
17. P. H. Zipkin, *Aggregation in Linear Programming*. Ph.D. Dissertation, Yale University (Dec. 1977).
18. A. Ali, R. Helgason, J. Kennington and H. Lall, *Solving multicommodity network flow problems*. Tech. Report IEOR 77015, Southern Methodist University (Nov. 1977).
19. J. K. Hartman and L. S. Ladson, A generalized upper bounding algorithm for multicommodity network flow problems. *Networks* 1, 333–354 (1972).
20. C. L. Liu, *Introduction to Combinatorial Mathematics*. McGraw-Hill, New York (1968).
21. J. Edmonds and R. M. Karp, Theoretical improvements in algorithmic efficiency for network flow problem, *J. Ass. Computing Mach.* **19**, 184–192 (1972).