Opl Res. Q., *Pergamon Press* 1975. *Vol.* 26, 4, ii, *pp.* 871 *to* 874. *Printed in Great Britain*

Bottleneck Linear Programming

A. M. FRIEZE
Queen Mary College, London University

We consider here the problem of finding a non-negative solution to a set of linear equations which minimizes a bottleneck type objective. Two algorithms are described for the general problem and a single algorithm for the bottleneck transportation problem.

INTRODUCTION

WE CONSIDER here the following optimization problem P

$$\text{minimize} \quad z = \max_{(j|x_j>0)} c_j$$

subject to

$$Ax = b,$$

$$x \geqslant 0,$$

where A, x, b and $c = (c_1, \ldots, c_n)$ are respectively an $m \times n$ matrix, an n-vector, an m-vector and an n-vector.

We call P a bottleneck linear program as it generalizes the bottleneck transportation problem as described by Garfinkel and Rao[1] and Hammer[2]. We shall describe two algorithms for solving P which are natural generalizations of algorithms described in the former paper.

ALGORITHMS

There seem to be two natural approaches to solving P. The first involves trying to improve a current solution.

(i) *Primal algorithm*

Step 0. Find an arbitrary feasible solution x to P.

Step 1. Let $\alpha = \max(c_j \mid x_j > 0)$. Let A^α be the submatrix of A consisting of those columns j for which $c_j < \alpha$.

Step 2. Find a solution to

$$\left.\begin{aligned} A^\alpha y &= b, \\ y &\geqslant 0. \end{aligned}\right\} \qquad (1)$$

If (1) has no feasible solution then the current x is optimal. Otherwise "extend" y to a solution x by putting $x_j = 0$ if y does not have a component y_j. Go to step 1.

The algorithm is clearly finite and produces an optimal solution.

(ii) *Threshold algorithm*

This algorithm adapts phase 1 of the two-phase simplex algorithm so that if the current solution is not feasible the column entering the basis minimizes c_j for j non-basic.

Step 0. Assuming $b \geqslant 0$ introduce a full vector of artificials $\xi = (\xi_1, \ldots, \xi_m)$ to create the augmented set of equations

$$Ax + I\xi = b,$$
$$x, \xi \geqslant 0.$$

Let $\pi = (1, \ldots, 1)$ be the current phase 1 pricing vector.

Step 1. If the current basic solution (x, ξ) is feasible, i.e. if $\xi = 0$ then the current solution is optimal. Otherwise denoting the jth column of A by a_j we define $F = \{j \mid \pi a_j > 0\}$. If $F = \phi$ then P is infeasible, otherwise let

$$c_k = \min(c_j \mid j \in F).$$

Step 2. Introduce x_k into the basis and carry out the appropriate pivot. Update π and go to step 1.

The algorithm terminates in a finite number of steps as it is a possible realization of the phase 1 simplex algorithm. That the solution obtained is optimal can be seen as follows: let $\alpha = \max(c_j \mid x_j > 0)$ and consider the tableau prior to the first time a variable x_k with $c_k = \alpha$ enters the basis. Since $\pi a_j \leqslant 0$ for all non-basic x_j with $c_j < \alpha$ we see that there is no non-negative solution to $Ax = b$ with all non-zero components having $c_j < \alpha$.

We note that in many cases it would be unnecessary to introduce a full artificial basis. If slack and surplus variables have minimum c values then the composite phase 1 algorithm may be used.

This algorithm can be viewed as an implementation of the threshold algorithm of Edmonds and Fulkerson[3] for the "clutter" (N, F) where $N = \{1, 2, \ldots, n\}$ and F is the family of supports of basic feasible solutions to $Ax = b$.

We have not tested these algorithms but extrapolating from the experience of Garfinkel and Rao[1] one would expect the threshold algorithm to be most efficient.

We note that one can easily provide for a secondary optimization of the form: find the optimal solution to P that minimizes the linear function $\tilde{c}x$. Starting with the optimal basis for P we apply the simplex algorithm ignoring any column with c_j larger than the optimum value of z.

Conversely if the main optimization is to be to minimize $\tilde{c}x$ and the objective function in P is subsidiary, one can continue from the optimum basis for $\tilde{c}x$ and apply either algorithm using only variables with a zero reduced cost.

BOTTLENECK TRANSPORTATION PROBLEM

We end with an algorithm for the bottleneck transportation problem

$$\text{minimize} \quad z = \max_{\{(i,j)|x_{ij}>0\}} c_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = a_i, \quad i = 1, \dots, m, \tag{2}$$

$$\sum_{i=1}^{m} x_{ij} = b_j, \quad j = 1, \dots, n, \tag{3}$$

$$x_{ij} \geqslant 0.$$

Step 0. Introduce "dummy" variables y_{ij} and augment (2) and (3) to

$$\sum_{j=1}^{n} x_{ij} + \sum_{j=1}^{n} y_{ij} = a_i, \quad i = 1, \dots, m, \tag{4}$$

$$\sum_{i=1}^{m} x_{ij} + \sum_{i=1}^{m} y_{ij} = b_j, \quad j = 1, \dots, m, \tag{5}$$

$$x_{ij}, y_{ij} \geqslant 0.$$

Find a basic feasible solution to (4) and (5) using only (y_{ij}) as basic variables. Any convenient rule, e.g. the N. W. Corner rule, can be used. Calculate a price vector $(\mathbf{u}, \mathbf{v})$ for this basis via $u_i + v_j = 1$ if y_{ij} basic.

Step 1. If $\sum_i \sum_j y_{ij} = 0$ terminate, the current solution is optimal, otherwise let $c_{kl} = \min(c_{ij} \,|\, x_{ij}$ non-basic and $u_i + v_j > 0)$.

Step 2. If y_{kl} is basic, put $x_{kl} = y_{kl}$, make x_{kl} basic in place of y_{kl} and go to step 1. If y_{kl} is non-basic introduce x_{kl} into the basis using the normal rules of the stepping stone algorithm and update $(\mathbf{u}, \mathbf{v})$. Note that we have $u_i + v_j = 1$ if y_{ij} basic and $u_i + v_j = 0$ if x_{ij} basic. Go to step 1.

One proves convergence of this algorithm in a similar manner to that of the threshold algorithm.

In their paper Garfinkel and Rao describe a simple means of calculating a lower bound z^0 to the minimum value of z. One could use those x_{ij} for which $c_{ij} \leqslant z^0$ in trying to make a better initial solution in step 0.

We note that one can derive similar algorithms by adding variables x_{i0} to (2) and x_{0j} to (3) and then proceeding as the added variables are slack variables or as the outflow and inflow from a dummy source and sink.

CONCLUSION

We have described a natural generalization of the bottleneck problem to a general linear programming context. The algorithms described are simple modifications of the simplex algorithm and therefore should be easy to implement.

[1] R. S. GARFINKEL and M. R. RAO (1971) The bottleneck transportation problem. *Nav. Res. Logist. Q.* **18**, 465.
[2] P. L. HAMMER (1969) Time minimizing transportation problems. *Nav. Res. Logist. Q.* **16**, 345.
[3] J. EDMONDS and D. R. FULKERSON (1970) Bottleneck extrema. *J. Combinatorial Theory* **8**, 299.