LINEAR PROGRAMMING PROBLEMS WITH BOUNDED VARIABLES AND A SINGLE CONSTRAINT - SOLUTION AND ALGORITHM

by

Yoav Raz

Technical Report #304
December 1983
ABSTRACT

The solution for linear programming problems with bounded variables and a single linear combination constraint is given. Necessary and sufficient conditions for the existence of a feasible solution and for a bounded optimum are derived. This solution is used for constructing a simple $O(n)$ space and $O(n \log n)$ time algorithm where $n$ is the number of variables. The algorithm has been implemented successfully on a personal computer for problems with thousands of variables.
1. INTRODUCTION

Several efficient methods are known for the solution of linear programming problems (see [H]). This work deals with a special case where one constraint is of a linear combination of the unknowns and all other constraints are bounds of the unknowns. A characterization of a solution is given from which a simple algorithm is derived. This algorithm requires $O(n)$ space and $O(n \log n)$ time, where $n$ is the number of unknowns. The algorithm has been implemented successfully on a personal computer for problems with thousands of unknowns.

In Chapter 2 several definitions and notations are given. The solution for the problem is characterized and investigated in Chapter 3. A necessary and sufficient condition for the existence of a feasible solution is given in Chapter 4, and in Chapter 5 the condition for the existence of an unbounded solution is derived. The algorithm for an optimal solution is described in Chapter 6, and time and space analysis is given in Chapter 7.

2. DEFINITIONS AND NOTATIONS

In this paper the following linear programming problem (*) is considered:

\begin{align*}
(1) & \quad a_i \leq x_i \leq b_i, \quad -\infty \leq a_i < b_i \leq \infty, \quad 1 \leq i \leq n \\
(*) & \quad \sum_{i=1}^{n} c_i x_i = d, \quad c_i \geq 0 \\
(3) & \quad \max Z = \sum_{i=1}^{n} z_i x_i.
\end{align*}
Remark: \( c_i \geq 0 \) can be assumed without loss of generality. (Otherwise, substitute \( x_i' = -x_i \).)

**Definition 1:** A feasible solution of (*) is an n-tuple \( S = (x_1,x_2,\ldots,x_n) \) which for its components \( x_1,x_2,\ldots,x_n \) (*1) and (*2) hold. \( Z(S) \) denotes the value of \( Z \) while substituting the components of \( S \) in (3).

**Definition 2:** An optimal solution of (*) is a feasible solution \( S \) for which for any other feasible solution \( S' \), \( Z(S) \geq Z(S') \) holds.

**Definition 3:** Let \( -\infty < k < \infty \) and let

\[ \{1,2,\ldots,n\} = I_{<k} \cup I_{=k} \cup I_{>k} \]

where

\[ I_{<k} = \{ i \mid z_i/c_i < k, \ i \in \{1,\ldots,n\} \} \]

(If \( c_i = 0 \) then if \( z_i > 0 \) then \( z_i/c_i \overset{\text{def}}{=} \infty \).

if \( z_i < 0 \) then \( z_i/c_i \overset{\text{def}}{=} -\infty \).

if \( z_i = 0 \) then \( x_i \) can be removed from the problem.)

A k-solution is an n-tuple \( S(k) = (x_1,x_2,\ldots,x_n) \) where

\[ x_i = a_i \quad \text{if} \quad i \in I_{<k} \]

\[ a_i \leq x_i \leq b_i \]

with \( a_i < x_i < b_i \) for at least one \( i \) if \( i \in I_{=k} \)

\[ x_i = b_i \quad \text{if} \quad i \in I_{>k} \]

**Remark:** If (*2) holds for \( S(k) \) then it is a feasible solution.
3. THE OPTIMAL SOLUTIONS

Lemma 1: A feasible k-solution is optimal.

Proof: Let \( S(k) = (x_1, x_2, \ldots, x_n) \) be a feasible k-solution, and let
\( S' = (x'_1, x'_2, \ldots, x'_n) \) be any feasible solution.

\[
Z(S(k)) - Z(S') = \sum_{i=1}^{n} z_i(x_i - x'_i) = \sum_{i \in I < k} z_i(a_i - x'_i) + \sum_{i \in I = k} z_i(x_i - x'_i) + \sum_{i \in I > k} z_i(b_i - x'_i).
\]

By (*1) and by Definition 3:

\[
\begin{align*}
    z_i(a_i - x'_i) & \geq k \ c_i(a_i - x'_i) & \text{if} & \quad i \in I < k \\
    z_i(x_i - x'_i) & = k \ c_i(x_i - x'_i) & \text{if} & \quad I \in I = k \\
    z_i(b_i - x'_i) & \geq k \ c_i(b_i - x'_i) & \text{if} & \quad I \in I > k
\end{align*}
\]

Hence by (*2)

\[
Z(S(k)) - Z(S') \geq k \left( \sum_{i=1}^{n} c_i x_i - \sum_{i=1}^{n} c_i x'_i \right) = k(d-d) = 0.
\]

In the following it is shown that every optimal solution should be \( S(k) \) for some k.

Lemma 2: Let \( S = (x_1, x_2, \ldots, x_n) \), \( S' = (x'_1, x'_2, \ldots, x'_n) \) be two
feasible solutions, where
\( x_i = x'_i \) if \( i \neq j \) and \( i \neq k \).

Suppose that the following relations hold:

\[
\frac{z_j}{c_j} < \frac{z_k}{c_k}, \quad x_k < x'_k.
\]

Then

\[
Z(S) < Z(S').
\]
Proof:

(1) \[ Z(S') - Z(S) = z_j(x_j' - x_j) + z_k(x_k' - x_k). \]

By (*)

\[ c_jx_j + c_kx_k = c_jx_j' + c_kx_k' \]

or

(2) \[ -c_j(x_j' - x_j) = c_k(x_k' - x_k). \]

Substituting (2) in (1) we get:

(3) \[ Z(S') - Z(S) = c_k \left( \frac{z_k}{c_k} - \frac{z_j}{c_j} \right) (x_k' - x_k) > 0. \]

(Since \( c_k > 0 \) by (*2), and by the assumptions of the lemma.) □

**Lemma 3**: An optimal solution is a feasible k-solution for some k.

Proof: Suppose \( S = (x_1, x_2, \ldots, x_n) \) is an optimal solution not in the form of \( S(k) \). Then, there exist two indices \( j, k \) such that the following holds:

\[
\begin{align*}
  x_k &< b_k \\
  x_j &> a_j \\
  \frac{z_k}{c_k} &> \frac{z_j}{c_j}
\end{align*}
\]

Now, if we increase \( x_k \) and maintain the solution feasible, we get by Lemma 2 a new feasible solution \( S' \) for which holds:

\[ Z(S') > Z(S). \]
This contradicts the optimality of $S$. Such $S'$ can be constructed in the following way: If $c_k = 0$ or $c_j = 0$ then increase or decrease $x_k$ or $x_j$ respectively to increase $Z$. If not let

$$S' = (x'_1, x'_2, \ldots, x'_n)$$

$$x'_i = x_i \quad \text{if } i \neq j \text{ and } i \neq k.$$  

By (*2)

$$x'_j = \frac{1}{c_j} \left( d - c_k x'_k - \sum_{i \neq j, i \neq k} c_i x'_i \right)$$

Let $\alpha, \beta$ be any finite numbers satisfying

$$a_j \leq \alpha < x_j$$

$$x_k < \beta \leq b_k.$$  

Define

$$x'_j = \max \left( \frac{1}{c_j} \left( d - c_k \beta - \sum_{i \neq j, i \neq k} c_i x'_i \right), \alpha \right)$$

and let

$$x'_k = \begin{cases} \frac{1}{c_k} \left( d - c_k \alpha - \sum_{i \neq j, i \neq k} c_i x'_i \right) & \text{if } x'_j = \alpha \\ \beta & \text{if } x'_j > \alpha \end{cases}$$

Clearly, $\beta > x'_k > x_k$ and $\alpha < x'_j < x_j$ and $S'$ is feasible. Hence

$$Z(S') > Z(S).$$

Lemmas 1, 3 resulting in the following theorem:

**Theorem 1:** A feasible solution is optimal if and only if it is a $k$-solution for some $k$. 
The nature of $k$ values giving a feasible $k$-solution can be deduced from the following lemma:

**Lemma 4:** If $I_{<k_1} \neq I_{<k_2}$ and some $k_1$-solution $S(k_1)$ is feasible then no $k_2$-solution is feasible.

**Proof:** Let $S(k_1) = (x_1, x_2, \ldots, x_n)$; $S(k_2) = (x', x_2', \ldots, x'_n)$.

If $I_{<k_1} \neq I_{<k_2}$ then or $I_{<k_1} \subseteq I_{<k_2}$ or $I_{<k_2} \subseteq I_{<k_1}$ occurs.

Suppose $I_{<k_1} \not\subseteq I_{<k_2}$. Then $I_{<k_1} \cup I_{=k_1} \subseteq I_{<k_2}$ and

$I_{>k_1} = I_{=k_2} \cup I_{>k_2}$.

Hence since $S(k_1)$ is feasible

$$d = \sum_{i \in I_{<k_1}} a_i c_i + \sum_{i \in I_{=k_1}} x_i c_i + \sum_{i \in I_{>k_1}} b_i c_i$$

$$> \sum_{i \in I_{<k_2}} a_i c_i + \sum_{i \in I_{=k_2}} x_i c_i + \sum_{i \in I_{>k_2}} b_i c_i$$

$$> \sum_{i \in I_{<k_2}} a_i c_i + \sum_{i \in I_{=k_2}} x_i' c_i + \sum_{i \in I_{>k_2}} b_i c_i$$

and $S(k_2)$ is not feasible.

The case $I_{k_2} \not\subseteq I_{k_1}$ is proved in the same way.

**Corollary:** Let $S(k)$ be a feasible solution of (*).

1. If $I_{=k}$ is not empty then $S(k')$ is not feasible for any $k' \neq k$, and the dimension of the optimal solutions is $|I_{=k}|-1$.

$|I|$ is the cardinality of $I$. 


(2) If \( I = k \) is empty then the optimal solution is unique (dimension = 0) and for any
\[
\max_{i \in I < k} \left( \frac{s_i}{c_i} \right) < k' < \min_{i \in I > k} \left( \frac{s_i}{c_i} \right)
\]
there is a feasible \( k' \)-solution \( S(k') \) and \( S(k) = S(k') \).

4. A CONDITION FOR THE EXISTANCE OF A FEASIBLE SOLUTION

**Theorem 2:** A necessary and sufficient condition for the existence of a feasible solution is

\[
(**) \quad \sum_{i=1}^{n} c_i a_i \leq d \leq \sum_{i=1}^{n} c_i b_i.
\]

**Proof.** If \( S = (x_1, \ldots, x_n) \) is feasible then by (**1)

\[
d = \sum_{i=1}^{n} c_i x_i \quad \leq \quad \sum_{i=1}^{n} c_i b_i\]

If (**) holds then suppose first that \( a_i, b_i \) \( i = 1, \ldots, n \) are bounded. Let

\[
x_i = a_i + (b_i - a_i) \theta
\]

where

\[
\theta = \frac{d - \sum_{i=1}^{n} c_i a_i}{\sum_{i=1}^{n} c_i b_i - \sum_{i=1}^{n} c_i a_i}
\]
Clearly, \( 0 \leq \theta \leq 1 \) by (**) and hence
\[
 a_i \leq x_i \leq b_i 
\]

Moreover
\[
 \sum_{i=1}^{n} c_i x_i = \Sigma_{i=1}^{n} a_i \cdot \frac{\Sigma c_i b_i - \Sigma c_i a_i}{\Sigma c_i b_i - \Sigma c_i a_i} = \Sigma_{i=1}^{n} b_i - \Sigma_{i=1}^{n} a_i = d .
\]

Hence \( S \) is feasible.

If \( a_i = -\infty \) or \( b_i = \infty \) it is replaced (for every such \( i \)) by finite \( a_i', b_i' \) respectively maintaining (**), and then \( S \) is calculated as in the bounded case.

5. A CONDITION FOR THE EXISTENCE OF AN UNBOUNDED OPTIMAL SOLUTION

Theorem 3: A necessary and sufficient condition for an unbounded optimal solution is that at least one of the following conditions is true:

(1) \( a_i = -\infty \) and \( z_i < 0 \), \( c_i = 0 \) \( \left( \frac{z_i}{c_i} = -\infty \right) \)

(2) \( b_i = \infty \) and \( z_i > 0 \), \( c_i = 0 \) \( \left( \frac{z_i}{c_i} = +\infty \right) \)

(3) \( a_k = -\infty \) \( b_j = \infty \) and \( \frac{z_j}{c_j} \geq \frac{z_k}{c_k} \).

Proof: If there is an unbounded feasible solution then \( a_i = -\infty \) or \( b_i = \infty \) for some \( i \).

If (1) (2)) holds then \( x_i \) can be unboundedly increased (decreased) while increasing \( Z \) and maintaining feasibility.
If (3) holds then letting $x_j$ go to $+\infty$ and $x_k$ to $-\infty$ while maintaining feasibility, the resulting $Z$ is going to $+\infty$ if $z_j/c_j > z_k/c_k$ (by Lemma 2), or $Z$ is remaining constant if $z_j/c_j = z_k/c_k$.

If $S$ is an unbounded optimal solution it is a feasible $k$-solution for some $k$, with one component at least unbounded.

If $x_j = +\infty (= b_j)$ then for the sake of feasibility or $c_j = 0$ or there is $x_k = -\infty (= a_k)$. If $c_j = 0$ then $z_j > 0$ in order to satisfy $z_j/c_j > k$, and (2) holds. If $x_k = -\infty (= a_k)$ then $j \in I_{>k}$ or $j \in I_{=k}$, $k \in I_{<k}$ or $k \in I_{=k}$ implying $z_j/c_j \geq k \geq z_k/c_k$ and hence (3) holds.

The case $x_k = -\infty$, $c_k = 0$ is treated in a similar way to that with $x_j = +\infty$, $c_j = 0$ implying (1).

6. AN ALGORITHM FOR AN OPTIMAL SOLUTION

The algorithm described here checks for the existence of a feasible solution (using Theorem 2) and then, if there is no unbounded optimal solution (using Theorem 3) builds a feasible $k$-solution. This solution is optimal by Theorem 1.

This algorithm can be implemented efficiently, totally in the main storage of a computer, or by using external storage for very large problems. In the second case, it can be implemented efficiently using only sequential processing of sequential files.

Definition 4: Let a file be defined as a sequence of tuples and denoted $((y_{1i}, y_{2i}, \ldots, y_{mi}) | i = 1, 2, \ldots, n)$. 
The Algorithm:

1. Input the parameters of (*);
2. if there is no feasible solution then stop;
3. sort \( (a_i, b_i, c_i, z_i/c_i) | i = 1, \ldots, n \) in descending order of \( z_i/c_i \) resulting in \( (a_{i_j}, b_{i_j}, c_{i_j}, z_{i_j}/c_{i_j}) | j = 1, \ldots, m \);
4. if there is an unbounded optimum then stop;
5. \( x_i = a_i \) for \( i = 1, 2, \ldots, n \); \( d' = \sum_{i=1}^{n} c_i a_i; j = 0 \);
6. while \( d' < d \) do
   begin
     \( j = j + 1 \);
     \( x_{i_j} = b_{i_j} \);
     \( d'' = d' \);
     \( d' = d' + c_{i_j} (b_{i_j} - a_{i_j}) \);
     end;
8. output \( x_i \) for \( i = 1, 2, \ldots, n \);
     stop;

Remarks:

1. If \( c_i = 0 \) then \( z_i/c_i \) should be given a sufficient high (low) value provided \( z_i \) is positive (negative).
2. If \( a_i = -\infty \) then it should be replaced before step 5 with a finite (small) \( a_i \) satisfying \( d' < d \).
   This alternative finite \( a_i \) should not be in the final solution \( (x_1, x_2, \ldots, x_n) \) at the end of step 6. The dropping of such \( a_i \) can be guaranteed by letting \( a_i \) to be small enough, but not too small in order to avoid a truncation error in step 6.
3. The finiteness of \( b_{i_j} \) in step 6 is guaranteed by step 4, because every such \( b_{i_j} \) appears in the resulting solution.
4. A division by $c_i = 0$ in step 6 is impossible since for $c_{ij}$ does not change in the loop.

5. Step 2 guarantees that in step 6 j does not exceed n.

Theorem 4: The algorithm proposed finds an optimal solution of (*).

Proof: If there is an unbounded optimum the algorithm stops in step 4.

In steps 5 and 6 a k-solution is calculated. The last line in step 6 guarantees the feasibility of the k-solution. Hence the solution is optimal by Theorem 1.

7. SPACE AND TIME ANALYSIS

Clearly, the algorithm can be carried out in $O(n)$ space, since for all subscripted data the subscripts are from 1 to n.

All the steps of the algorithm, but step 3, can be processed in time $O(n)$. Step 3 requires $O(n \log n)$ time. Hence the overall time for the algorithm is $O(n \log n)$.

Remark: The author suspects that the solution of this simple problem is not new, but since he could not find any evidence for this he finds it useful and interesting enough to have it published.

REFERENCE