1996

The simplex algorithm for linear programming is of polynomial complexity

Roger N. Pederson
Carnegie Mellon University

The Simplex Algorithm for Linear Programming is of Polynomial Complexity

by

R. N. Pederson

Department of Mathematics
Carnegie Mellon University
Pittsburgh, PA 15213

Research Report No. 96-190
June, 1996

The Simplex Algorithm for Linear Programming is of Polynomial Complexity

R. N. Pederson

1 Introduction.

It is the purpose of this note to show that the Simplex Algorithm as stated in [1] page 5 is of polynomial complexity as long as redundant constraints, when they arise, are removed. In Section 2, we show this applies to the Feasible Set part of the algorithm. Finally, in Section 3 we show how the proof should be modified when all of the coefficients are positive.

We tacitly assume that a degenerate vertex is never encountered; these can be handled as in Section 4 of [1].

2 The Feasible Set Algorithm.

Let us consider the simplex algorithm in the notation (1.1), (1.2) of [1] with the additional assumption that $\mathcal{A}_{i,n+1} > 0$, $n+1 < i < p$ and $\mathcal{A}_{p,n+1} < 0$. We shall show that at most n updating steps leads to a redundant constraint or $\mathcal{A}_{p,n+1} > 0$.

The key to our proof is the expression of the matrix coefficients after r updating steps in terms of the minor determinants (.8.7),...,(.7.10). In addition we may avoid the reasoning that led to (14.14.12) and deduce them from the fact that the next step leads to a non-empty set with no redundant constraints, using the fact that $f_2\,(i,p:q,\ell) < 0$, $f\,(j,p:g,\ell)$ are incompatible when $\ell \in S_i^- \cap S_p^+$, $q \in S_i^+ \wedge S_p^-$ and that the latter fact must arise as a consequence of (13.11)–(13.13) of [1].

Let us then take as our induction hypotheses the assumption that after r updating step, starting with $r = 0$, the elements (.8.10) of [1] are all negative and, suppressing the dependence on $i_1, \ldots, i_i, \ 1, \ldots, r$,

$$(-1)^{r+1} f_{r+1}(i_{r+1}, p : r+1, r+2) < 0 \tag{1}$$

and

$$(-1)^{r+1} f_{r+1}(i_{r+1}, p : r+1, r+2) > 0. \tag{2}$$

The condition (2.1) together with the consequence

$$sqn\{D_r\} = (-1)^r \tag{3}$$

of the recursion formula (.10.12) insure that

$$\mathcal{A}^r_{i_{r+1}, j_{r+1}} > 0 \tag{4}$$

this being obtained after the interchange of the basic constraint indexed by i_{r+1} and $r+1$. Similarly, if we would have interchanged the constraint indexed by i_{r+2} with $(r+1)^{\text{st}}$, we would obtain the androgenous result to (2.4). We note that Theorems (8.1) and (12.2) insure that insure that no two k_r's are equal for the contrary would violate the fact that each updating step increases the constant of the p^{th} constraint. Now we notice from (8.10) and (8.2) of [1] that

$$\mathcal{A}^{r-1}_{i_n, r} = \mathcal{A}^r_{i,r} D_r D_{r-1}. \tag{5}$$

Hence, they have oppositve signs. Hence, the list corresponding to the $(i+1)^{\text{st}}$ step has at least one element in the list corresponding to (8.8) of [1]. The element corresponding to (2.2) is not in the present list array but if we had instead interchanged the constraints indexed by c_{r+1} and $r+2$. We would have obtained one more. By applying the invariance theorem to the appropriate permutation we would deduce that there are now elements of (8.10) are now all negative with r increased by one. This completes the proof by induction that at most n updating steps produces either a redundant constraint or a complete simplex step.

3 The General Case.

In this case, we use the strategy of Section 7 of [1] to eliminate a variable. Then we are back to the Feasible Set Case for finding the maximum on a face. This shows the additional advantage of eliminating as many redundant constraints as possible, but counting faces rather than vertices greatly reduces the maximum complexity. Also, we may start the elimination process by taking the constraint that is closest to being parallel to the objective function. While the choice then is not invariant under changes of variables, it should tend to reduce the complexity.

References

[1] Pederson, R. N., *New Methods in the Simplex Method for Linear Programming*, Raofest, Dekker.

[2] Aspvall, Bengt and Stone, Richard, Khachiyan's Linear Programming Algorithm, *Journal of Algorithms,* **1** (1986), pp. 1-13.

Acknowledgement. I would like to thank Ross Willard for pointing out Reference [2] to me.