



*Journal of Advances in
Science and Technology*

*Vol. VI, Issue No. XII,
February-2014, ISSN 2230-
9659*

A CRITICAL STUDY ON BOUNDS AND LOWER OPTIMIZATION

AN
INTERNATIONALLY
INDEXED PEER
REVIEWED &
REFEREED JOURNAL

A Critical Study on Bounds and Lower Optimization

Satpal Singh

Assistant Professor, Dept. of Mathematics, University College, Kurukshetra (Haryana) Pin Code - 136118

Abstract – Linear optimization can be used for set of different problems, which have restrictions such as a given amount of resources or a certain budget. The lower bound is the smallest possible value, and the upper bound is the highest possible value. Linear programming is the most commonly used optimization technique in embedded industrial applications. There are many reasons for this. Linear programs are relatively easy to formulate, use and understand.

INTRODUCTION

The LP optimization techniques are also relatively efficient and well developed. A surprisingly large set of real-life problems can be represented as linear programs, or approximated sufficiently well with linear programs. Finally, a number of more advanced modeling and solution techniques are based on linear programming, such as quadratic programming, fractional programming, integer programming, mixed integer programming, constraint logic programming, etc.

Deterministic algorithm for solving AOPs must in the worst case examine all bases of the input AOP to find the optimal one. This implies that the randomization used in the previous chapter to obtain sub-exponential bounds was indeed crucial. In other words, while the sub-exponential algorithm can only 'fool' itself by coming up with 'bad' coin flips, any deterministic algorithm can be fooled by an adversary who will supply just the problem the algorithm cannot handle efficiently.

Theorem Let H be a finite set, $B \subseteq 2^H$ a set of bases. For any deterministic algorithm A , there exists an ordering $\leq$ of the bases and an improvement oracle ϕ such that A needs at least $|B| - 1$ oracle queries to solve $(H, B, \leq \Phi)$.

Proof. We start A on some problem $(H, B, \leq \Phi)$ with $\leq$ and ϕ not yet determined, and we argue that an adversary answering the oracle queries can construct $\leq$ and ϕ 'online' in such a way that the algorithm is forced to step through all the bases. When supplied with a query pair (G, B) , the adversary will output an answer $B' = \phi(G, B)$ according to two simple rules:

(i) the answer B' is consistent with the previous ones, i.e. there exists an AOP on H and B such that

the present and all previous queries have been answered correctly with respect to this AOP.

(ii) The answer $B' = B$ (proving that B is optimal among all subsets of G) is given only if there is no other consistent answer.

It is clear by induction that the adversary always has a consistent answer, so the algorithm A steps through a sequence of queries with pairs (G, B) and finally stops. Suppose that less than $|B| - 1$ queries have been performed. Then there are two bases B_1 and B_2 which have never been the second component of a query pair. We will show that it is consistent with all answers to assume that $B_1 = B(H)$, i.e. B_1 is an optimal basis.

The same holds for B_2 , so whatever the algorithm outputs, there is an AOP that is not correctly solved, a contradiction to A being an algorithm for solving any AOP. We are left to prove that $B_1 = B(H)$ is a consistent choice. Clearly, this can fail only if some answer has revealed the existence of a larger basis with respect to $\leq$. Since there was no query pair (G, B_1) , the only remaining possibility for this to happen is that some query (G, B) with $B_1 \subseteq G$ has been answered by B , thus establishing $B_1 \subseteq B$. But in the first query of this type, B_1 was still 'free' and could have been returned instead of B , a contradiction to rule (ii).

REVIEW OF LITERATURE

The bases of LP-type systems as well as the z -values can be identified with $GF(2)^{\delta}$, the vector space of all $0=1$ -vectors of length δ over the two-element field $GF(2)$. This, on the one hand, allows to reinterpret algorithms as flipping games over $GF(2)^{\delta}$ and on the other hand makes linear algebra over $GF(2)$ available as a very useful tool for the analysis of these games.

Let

$$H := \{h_1, \dots, h_\delta\}$$

$$\bar{H} := \{\bar{h}_1, \dots, \bar{h}_\delta\}$$

be two disjoint element sets. For any l , h_l and $\bar{h}_l$ are called conjugate elements. We consider LP-type systems

$$L_A = (H \cup \bar{H}, B, z_A)$$

$$B := \{B \subseteq H \cup \bar{H} \mid |B \cap \{h_l, \bar{h}_l\}| = 1, \text{ for all } l\}$$

i.e. the bases are all the sets containing exactly one of any two conjugate elements. Every basis has δ elements and can be identified with a 0/1 vector $\vec{B}$ whose l -th component is defined by

$$\vec{B}_l := \begin{cases} 0, & \text{if } h_l \in B \\ 1, & \text{if } \bar{h}_l \in B \end{cases}$$

The mapping $B \rightarrow \vec{B}$ defines a bijection between B and $GF(2)^\delta$.

$$A = \begin{pmatrix} a_{11} & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ a_{\delta 1} & \dots & \dots & a_{\delta\delta} \end{pmatrix} \in GF(2)^{\delta \times \delta}$$

with $a_{ll} = 1$ for all l (equivalently, A is nonsingular). The objective function z_A assigns to any basis B the vector

$$z_A(B) := A\vec{B} \in GF(2)^\delta$$

where $GF(2)^\delta$ becomes an ordered set W via decreasing lexicographic order, i.e. a vector is larger than another if it has a zero-entry in the first component where they differ. This agrees with the usual decreasing order of the natural numbers when 0=1-vectors are interpreted as binary numbers in the obvious way. The reason for using decreasing instead of the natural increasing order is merely technical and will become clear soon. Since A is nonsingular, all the bases get distinct values and can uniquely be ordered according to these values.

Example Let $\delta = 3$,

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

The following table lists the 0/1-vectors corresponding to the bases, ordered by increasing z_A -value.

$\vec{B}$	$z_A(B) = A\vec{B}$	binary value
100	111	7
101	110	6
111	101	5
110	100	4
010	011	3
011	010	2
001	001	1
000	000	0

Thus, the all-zero vector is the optimal basis in all problems L_A . The following lemma prepares the proof that L_A indeed defines an LP-type system. Recall that z_A extends to all subsets G of H $z_A(G) := \max \{z_A(B) \mid B \in \mathcal{B}, B \subseteq G\}$.

Lemma for any $G \subseteq H \cup \bar{H}$, $B \subseteq G$ basis, the following are equivalent.

- (i) $B = \mathcal{B}(G)$, i.e., $z_A(B) = z_A(G)$.
- (ii) $z_A(B)_l = 0$ for all l with $h_l, \bar{h}_l \in G$.

Proof. Assume (i) holds and consider l with $h_l, \bar{h}_l \in G$. Then the basis

$$B' := B \Delta \{h_l, \bar{h}_l\}$$

is in G as well, with

$$z_A(B') = A\vec{B}' = A(\vec{B} + e_l) = z_A(B) + A_{e_l},$$

where A_{e_l} coincides with the l -th column of A .

Now suppose (ii) is satisfied. Choose any basis $B' \subseteq G, B' \neq B$ and let l be minimal such that $\vec{B}_l \neq \vec{B}'_l$. Then $h_l, \bar{h}_l \in G$, so

$$z_A(B)_l = 0$$

Moreover, because of A's shape, l is also minimal with

$$z_A(B)_l = (\vec{AB})_l \neq (\vec{AB}')_l = z_A(B')_l,$$

and this shows $z_A(B) > z_A(B')$. Since B' was any basis, $B = B(G)$ follows.

Theorem For any lower-diagonal, nonsingular matrix $A \in GF(2)^{\delta \times \delta}$

$$L_A = (H \cup H, B, z_A)$$

Proof. z_A extends to $2^{H \cup R}$ is automatically satisfied if no basis contains another which is the case because all bases have the same size. Condition (3.2) demands that

$$z_A(G \cup \{j\}) > z_A(G) \text{ implies}$$

$$z_A(B \cup \{j\}) > z_A(B),$$

RESEARCH METHODOLOGY

Recall that the primitive operations needed to solve an LP-type system by Algorithm are the improvement query,

$$z_A(B \cup \{j\}) > z_A(B)?$$

and the basis computation,

$$B' := B(B \cup \{j\})$$

Actually, a basis improvement primitive suffices but in this case we do have a basis computation available. The following lemma shows what the operations look like when we interpret them as operations on the 0/1-vector $z_A(B) = \vec{AB}$ rather than on B itself. This reinterpretation subsequently leads to an equivalent formulation of algorithm as a flipping game on objective function values.

Lemma: Consider a basis $B, j \in \{h_l, \bar{h}_l\}, j \notin B$. Then

$$z_A(B \cup \{j\}) > z_A(B) \text{ if and only if } (\vec{AB})_l = 1$$

and

$$B' = B(B \cup \{j\}) \text{ if and only if } \vec{AB}' = F_A^l \vec{AB},$$

where

$$F_A^l := E + \left(\underbrace{0, \dots, 0}_{l-1}, A_{el}, 0, \dots, 0 \right) \in GF(2)^{\delta \times \delta}$$

is called the flip at l .

Proof. Let us distinguish two cases. If $B' = B$, then $B' = B(B \cup \{j\})$ equivalently means $z_A(B \cup \{j\}) = z_A(B)$, resp. $(\vec{AB})_l = 0$. This in turn holds if and only if

$$(0, \dots, 0, A_{el}, 0, \dots, 0) \vec{AB} = 0, \text{ equivalent ly } \vec{AB}' = \vec{AB} = F_A^l \vec{AB}.$$

If $B' \neq B$, then $B' = B(B \cup \{j\})$ if and only if $(\vec{AB})_l = 1$, where $B' = B \Delta \{h_l, \bar{h}_l\}$ (this is the only other basis in $B \cup \{j\}$). But then we equivalently get $(\vec{AB})_l = 1$ and

$$\vec{AB} = A(\vec{B} + e_l) = \vec{AB} + A e_l = \vec{AB} + (0, \dots, 0, A_{el}, 0, \dots, 0) \vec{AB} = F_A^l \vec{AB}$$

To make this convenient interpretation of basis computations as improving linear transformations of objective function values work, decreasing lexicographic order has been chosen (no linear transformation can improve on the value 00 ... 0, therefore this has to be the optimum value).

$$F_A^1 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}, F_A^2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}, F_A^3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

and the step from basis B to B' with

$$\vec{B} = 100, \vec{B} + e_2 = 110 \text{ takes } z_A(B) = \vec{AB} = 111$$

$$z_A(B') = A\vec{B}' = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = F_A^2 A\vec{B}$$

CONCLUSION:

In general, when the flip at l is applied to a vector whose l -th entry has value one, it zeroes this entry and flips values at any other position where Ae_l equals one. Since A was lower-diagonal, all these positions are to the right of l , so the vector becomes lexicographically smaller. Note that whether a value to the right of l gets flipped does not depend on the value itself but only on its position in the vector. Moreover, if the flip at l is applied to a vector whose l -th entry has value zero, this vector remains unchanged. Algorithm RF-LPtype works on pairs (G, B) with $B \subseteq G \subseteq H \cup H$, B basis. The procedure RF-FlipA equivalently represents B by

$V(B) := AB^+$ and G by the set

$$S(G) := \{l \in G\} \mid \{h_l, \bar{h}_l\} \subseteq G\}$$

$S(G)$ contains exactly the subscripts that appear in $G - B$ but does not keep the information whether h_l or its conjugate $\bar{h}_l$ is in $G - B$. However, given B (which uniquely corresponds to $V(B)$), this information is deductible and G can be recovered Invoking Lemma.

REFERENCES:

- **B. Gartner.** A subexponential algorithm for abstract optimization problems. In Proceedings of 33rd Annual IEEE Symposium on Foundations of Computer Science.
- **B. Gartner and E. Welzl.** Vapnik-Chervonenkis dimension and pseudo-hyperplane arrangements. Discrete Comput. Geom.
- **B. Gartner and E. Welzl.** Linear programming randomization and abstract frameworks. In Proceedings of 13th Annual Symposium on Theoretical Aspects of Computer Science (STACS), volume 1046 of Lecture Notes in Computer Science.
- **K. L. Clarkson.** Linear programming in $O(n^3 d^2)$ Time. Information Processing Letters.
- **K. L. Clarkson.** A Las Vegas Algorithm for Linear and Integer Programming when the Dimension is Small. In Proceedings of 29th Annual IEEE Symposium Foundations of Computer Science. (1988).
- **T. H. Cormen, C. E. Leiserson, and R. L. Rivest.** Introduction to Algorithms. The MIT Press, Cambridge, MA.
- **G. B. Dantzig.** Linear Programming and Extensions. Princeton University Press, Princeton, NJ.
- **M. E. Dyer.** On a Multidimensional Search Technique and Its Application to the Euclidean One-Center Problem. SIAM J. Comput.
- **M. E. Dyer and A. M. Frieze.** A Randomized Algorithm for Fixed-Dimensional Linear Programming. Math. Programming.
- **H. Edelsbrunner and R. Seidel.** Voronoi diagrams and arrangements. Discrete Comput. Geom.