Polynomial Algorithms for LP over a Subring of the Algebraic Integers with Applications to LP with Circulant Matrices

Ilan Adler and Peter A. Beling
Department of Industrial Engineering and Operations Research
University of California, Berkeley
Berkeley, CA 94720

Abstract

We show that a modified variant of the interior point method can solve linear programs (LPs) whose coefficients are real numbers from a subring of the algebraic integers. By defining the encoding size of such numbers to be the bit size of the integers that represent them in the subring, we prove the modified algorithm runs in time polynomial in the encoding size of the input coefficients, the dimension of the problem, and the order of the subring. We then extend the Tardos scheme to our case, obtaining a running time which is independent of the objective and right-hand side data. As a consequence of these results, we show that LPs with real circulant coefficient matrices can be solved in strongly polynomial time. Finally, we show how the algorithm can be applied to LPs whose coefficients belong to the extension of the integers by a fixed set of square roots.

1 Introduction

The question of whether a linear programming problem can be solved in polynomial time was answered in a landmark paper by Khachiyan in 1979. In fact, both Khachiyan's ellipsoid method [7] and Karmarkar's interior point method [6] solve linear programs (LPs) with rational coefficients in time that is polynomial in the number of input coefficients and the total number of bits in a binary encoding of the problem data. Nevertheless, several interesting questions concerning the complexity of linear programming remain open. One of the main open questions is usually stated as: Is there a strongly polynomial algorithm for linear programming? Following standard usage (cf. [15]), we say an algorithm for linear programming is *strongly polynomial* if

(S1) it consists of the elementary operations addition, subtraction, multiplication, division, and comparison,

(S2) the total number of elementary operations performed by the algorithm is polynomial in the number of input items (i.e., the number of coefficients in the matrices and vectors that define the input), and

(S3) *when applied to a rational instance*, the (binary encoding) size of the numbers that occur during the course of the algorithm is bounded by a polynomial in the size and total number of input items.

The conditional nature of (S3) stems from differences between rational number and real number models of computation. The standard model of computation for rational numbers is derived from the Turing machine. Consequently, the time required to perform an elementary arithmetic operation (S1) depends on the bit size of the operands. Requirement (S3) ensures that the time the algorithm spends performing any intermediate calculation is polynomial in the input size. In the case of real numbers, linear programming is usually modeled in terms of a machine that can perform any of the elementary operations (S1) in constant time, regardless of the nature of the operands. (See [2] for a rigorous treatment of general computation with real numbers.)

Taking advantage of the dichotomy implied by (S3), we can split the question of the existence of a strongly polynomial algorithm into two easier questions:

(A) If the data is rational, does there exist an algorithm satisfying (S1), (S2), and (S3)?

(B) If the data is real, does there exist an algorithm satisfying (S1) and (S2)?

To date, efforts to find a strongly polynomial algorithm for linear programming follow one of two main approaches, distinguished by whether they are directed at question (A) or at question (B). Those efforts directed at question (A) involve modifying existing polynomial-time algorithms, such as the ellipsoid method or variants of the interior point algorithm, so that their running times become independent of the size of at least part of the input data. In a key result along these lines, Tardos [15] showed that a LP can be solved in time that is independent of the size of the data in the objective and right-hand side vectors. As a consequence, LPs in which the coefficient matrix has 'small' rational entries, such as those that arise frequently in combinatorial optimization, can be solved in strongly polynomial time. Recently, Norton, Plotkin, and Tardos [13] extended Tardos' results by giving an algorithm whose running time is independent of the size of the data in a fixed number of rows or columns of the coefficient matrix.

Before discussing work directed at question (B), we mention some issues concerning the existing polynomial algorithms for rational LPs. Both the ellipsoid and interior point methods depend in a fundamental way on upper and lower bounds on the magnitude of certain numbers related to basic solutions of the LP. These bounds allow the algorithms to be properly initiated and terminated, and are themselves part of the theoretical complexity of the algorithms. If the problem is rational, the bounds are a function of the bit size of the problem data and can be computed in polynomial time. If the problem is not rational, it is still possible to compute the necessary upper bounds in polynomial time, but no polynomial method for computing the lower bounds is known [10].

The second approach toward finding a strongly polynomial algorithm for linear programming focuses on answering question (B) for special classes of LPs. Considering the discussion above, it is not surprising that these efforts generally involve the construction of algorithms that are greatly different from the existing polynomial algorithms for rational data. The work of Megiddo provides important examples of this second approach; in [8] a strongly polynomial algorithm is given for feasibility problems in which at most two variables appear in each inequality, and in [9] a strongly polynomial algorithm is given for problems in which the number of variables is fixed. Interestingly, the combination of the ideas in this latter algorithm with those in [15] led to the algorithm in [13].

In this paper, we show that linear programs whose coefficient matrices are circulant can be solved in strongly polynomial time. (LPs of this kind are related to discrete convolution and arise frequently in image processing.) In proving this result, we are led to an analysis of polynomial-time algorithms for linear programming in a subring of the algebraic integers. Specifically, we show that a variant of the interior point method can solve LPs whose coefficients are real members of the set

$$W_p = \{\alpha : \alpha = \textstyle\sum_{j=0}^{p-1} a_j \omega^j \,; a_j \text{ integer } \forall j\},$$

where $\omega = e^{2\pi i/p}$ is the first primitive p^{th} root of unity. (We lose no generality by working with the subring W_p instead of the subfield $\Omega_p = \{\gamma : \gamma = \sum_{j=0}^{p-1} q_j \omega^j \,; q_j \text{ rational } \forall j\}$.) The restriction to W_p allows us to define the 'encoding size' of the input number α to be the bit size of the integers $a_0, \ldots, a_{p-1}$ in the representation $\alpha = \sum_{j=0}^{p-1} a_j \omega^j$.

The key to our construction is our ability to obtain 'reasonable' upper and lower bounds on certain quantities involving the basic solutions of the LP. These bounds are a function of p (the order of the subring in which we work) and the encoding size of the data. We use these bounds to show that the modified algorithm's running time is polynomial in the dimension of the LP, the order p of the subring, and the encoding size of the data. We then proceed to modify the scheme given by Tardos [15] for rational data so that it works with data from W_p. The modified Tardos scheme gives us an algorithm whose running time is independent of the encoding size of the objective and right-hand side data (in fact, the objective and right-hand side vectors are allowed to be arbitrary real numbers).

Finally, we show that the numbers belonging to the extension of the integers by a set of positive integer square roots are embedded in W_p, for some known p. Using our earlier results, we then obtain an algorithm to solve LPs with such coefficients in time that is polynomial in the problem dimension, the encoding size of the input data, and the product of all the square roots.

The paper is organized in the following manner: In section 2, we discuss circulant matrices and show that LPs with a circulant coefficient matrix can be polynomially transformed into an equivalent LP in which the entries of the coefficient matrix are small in magnitude and belong to W_n, where n is the dimension of the matrix. We also show that if the original data is rational then these entries are integers. In this case, the Tardos scheme [15] gives us a strongly polynomial algorithm directly. In section 3, we analyze general LPs whose coefficients belong to W_p. In particular, we modify a variant of the interior point algorithm to solve these problems in polynomial time. In section 4, we use the results of section 3 and the scheme in [15] to obtain an algorithm that runs in time independent of the encoding size of the objective and right-hand side data. In section 5, we show how our results can be applied to LPs whose coefficients belong to an extension of the integers by a set of square roots. Finally, in section 6, we conclude with some remarks.

2 Linear programming with circulant matrices

In this section we bound the complexity of linear programs whose coefficient matrices are circulant.

Definition. The $p \times p$ matrix A (possibly with complex entries) is said to be *circulant* if and only if it has the following form:

$$A = \begin{bmatrix} a_0 & a_1 & \cdots & a_{p-2} & a_{p-1} \\ a_{p-1} & a_0 & \cdots & a_{p-3} & a_{p-2} \\ \vdots & & \ddots & & \vdots \\ a_1 & a_2 & \cdots & a_{p-1} & a_0 \end{bmatrix}.$$

Given a vector $a^T = (a_0, \ldots, a_{p-1})$, we shall use circ (a) to denote the circulant matrix defined above. (If M is a matrix or vector, we use M^T to denote the transpose of M.)

Before focusing on linear programing, we need to develop a number of the properties of circulant matrices. We begin along these lines by defining a matrix with the interesting property that it diagonalizes every circulant matrix.

Definition. Let ω be the first primitive p^{th} root of unity; that is,

$$\omega = e^{2\pi i/p}, \text{ where } i = \sqrt{-1}.$$

Then we define F_p to be the $p \times p$ matrix whose jk^{th} component is $\omega^{(j-1)(k-1)}$.

The matrix F_p is usually called the p^{th} order *Fourier matrix*, the label Fourier coming from the fact that the product $F_p a$ represents the discrete Fourier transform of the p-vector a. By using the identity $\omega^j = \omega^{j \bmod p}$ and by manipulating geometric series, it is easy to show that $\sqrt{1/p}\,F_p$ is unitary; that is, $F_p^{-1} = (1/p)F_p^*$, where F_p^* denotes the Hermitian transpose of F_p (the jk^{th} component of F_p^* is $\omega^{(1-j)(1-k)}$). In an effort to keep notation simple, we omit the index p on F_p when the size of the Fourier matrix is clear from the context of the discussion.

We now state several well-known results on circulant matrices. Detailed proofs of these results can be found in [3].

Proposition 2.1 *(i) Let A = circ (a) be a $p \times p$ circulant matrix. Then the columns of the p^{th} order Fourier matrix F are eigenvectors of A and the entries of the p-vector Fa are the corresponding eigenvalues. (ii) A is a $p \times p$ circulant matrix if, and only if, $A = (1/p)FGF^*$ for some $p \times p$ diagonal matrix G.*

Proposition 2.2 *Let G be a $p \times p$ diagonal matrix and let $A = (1/p)FGF^*$. Then, the pseudoinverse (sometimes also called the generalized or Penrose-Moore inverse) of A is given by $A^+ = (1/p)FG^+F^*$, where G^+ is a $p \times p$ diagonal matrix with entries*

$$G^+_{jj} = \begin{cases} 1/G_{jj} & \textit{if } G_{jj} \neq 0 \\ 0 & \textit{otherwise.} \end{cases}$$

It follows from proposition 2.2 and part (ii) of proposition 2.1 that the matrix A is circulant if and only if its pseudoinverse is circulant.

We turn now to an analysis of linear programming with circulant matrices. Specifically, we consider the following standard form problem:

$$\begin{array}{lll} (P) & \min & c^T x \\ & \text{s.t.} & Ax \ = b \\ & & x \ \geq 0, \end{array}$$

where $a, b, c, x \in \mathbf{R}^n$ and A = circ (a). It is easy to see that if $Ax = b$ is consistent then (P) is equivalent to the following problem:

$$\begin{array}{lll} (P') & \min & c^T x \\ & \text{s.t.} & nA^+Ax \ = nA^+b \\ & & x \ \geq 0. \end{array}$$

Now, by proposition 2.2 and part (ii) of proposition 2.1, $A^+A = \{(1/n)FG^+F^*\}\{(1/n)FGF^*\}$. Using the identity $(1/n)F^*F = I$, we then have $nA^+A = FDF^*$, where D is a diagonal matrix with ones and zeros in positions on the diagonal that correspond to the positions of the nonzero and zero elements of G, respectively. By the definition of F and F^* and since $\omega^j = \omega^{j \bmod n}$, it is clear that each entry of the matrix nA^+A has the form $\sum_{j=0}^{n-1} d_j \omega^j$, where d_j is either 0 or 1.

We now note that it is easy to compute A^+ from A.

Proposition 2.3 *Let A be a given $n \times n$ circulant matrix. Then it is possible to compute A^+ from A using only Gaussian elimination and a constant number of matrix multiplications, each of order n. Moreover, if the entries in A are rational, then so are the entries in A^+.*

Next, we present a few standard results from number theory that will be of key importance when we derive complexity bounds for rational instances of (P).

Definition. A complex number α is called an *algebraic integer* if there exists integers $d_0, \dots, d_{k-1}$ such that $\alpha^k + d_{k-1}\alpha^{k-1} + \cdots + d_0 = 0$.

Note that because it satisfies the polynomial equation $\omega^n - 1 = 0$, the number $\omega = e^{2\pi i/n}$ is an algebraic integer by the above definition.

Proposition 2.4 *The set of algebraic integers is closed under addition, multiplication, and negation.*

Since ω is an algebraic integer, proposition 2.4 implies that if α has the form $\alpha = \sum_{j=0}^{n-1} d_j \omega^j$, where d_j is integer, then α is an algebraic integer.

Proposition 2.5 *An algebraic integer is rational if and only if it is an integer.*

Proofs of propositions 2.4 and 2.5 can be found in most texts on algebraic number theory (see, e.g., [5]).

If the coefficient matrix A in (P) is rational we can use propositions 2.4 and 2.5 to make a strong statement about the form of the entries in nA^+A, the coefficient matrix of (P'):

Proposition 2.6 *If A is an $n \times n$ circulant matrix with rational coefficients, then the entries of nA^+A are integers.*

Armed with the above results, we can easily bound the complexity of rational instances of (P):

Theorem 2.1 *Suppose that A is an $n \times n$ circulant matrix and that the entries of $A, b,$ and c are rational. Then (P) can be solved in strongly polynomial time.*

Proof. As a preprocessing step in the solution of (P), one can use Gaussian elimination to check the consistency of $Ax = b$. If $Ax = b$ is inconsistent, then obviously (P) is infeasible and no further work is required. If $Ax = b$ is consistent, then by proposition 2.3 one can convert (P) into the equivalent problem (P') using Gaussian elimination and matrix multiplication. In either case the dominant computational work is Gaussian elimination, which is a strongly polynomial operation (see [4]).

In the discussion preceding proposition 2.3 we established that every entry α in the coefficient matrix

nA^+A has the form $\alpha = \sum_{j=0}^{n-1} d_j\omega^j$, where d_j is either 0 or 1. Since $|\omega^j| = 1$ for all j, we see that $|\alpha| \leq n$. But because we assume that A is rational, α must be an integer by proposition 2.6. Thus, the coefficient matrix of (P') contains integers of absolute value at most n. Moreover, the right-hand side of (P') is rational; if (P) is rational then by proposition 2.3 A^+ is rational, hence so is A^+b.

Tardos [15] showed that LPs of the form (P') with rational coefficients can be solved in time polynomial in the problem dimension and the binary encoding size of the numbers in the coefficient matrix. Since the binary encoding size of nA^+A is polynomial in the problem dimension, (P'), and therefore (P), can be solved in strongly polynomial time. □

For the purpose of analyzing LPs with circulant coefficient matrices and real coefficients (including those in the objective and right-hand side), we adopt a model of computation that allows constant time addition, subtraction, multiplication, division, and comparison of real numbers. Under this model we have the following result:

Theorem 2.2 *Suppose that A is an $n\times n$ circulant matrix and that the entries of A, b, and c are real numbers. Then (P) can be solved in strongly polynomial time.*

To prove theorem 2.2 we need a number of new results concerning the numbers in nA^+A. These results, which we shall obtain in sections 3 and 4, lead in a natural way to an analysis of linear programming in a subring of the algebraic integers. The proof of theorem 2.2 follows directly from this analysis and is given at the end of section 4.

3 LP over a subring of the algebraic integers

We consider the following primal-dual pair of LPs:

$$
\begin{array}{llll}
(P) & \min & c^Tx & \\
& \text{s.t.} & Ax & = b \\
& & x & \geq 0
\end{array}
\qquad
\begin{array}{llll}
(D) & \max & b^Ty & \\
& \text{s.t.} & A^Ty + z & = c \\
& & z & \geq 0,
\end{array}
$$

where $A \in \mathbf{R}^{m\times n}, b \in \mathbf{R}^m$, and $c, x \in \mathbf{R}^n$. Throughout this section we assume that all entries of A, b, and c are real and belong to the following set:

$$W_p = \left\{ \alpha : \alpha = \sum_{j=0}^{p-1} a_j\omega^j ; a_j \in \mathbf{Z}\ \forall j \right\},$$

where $\omega = e^{2\pi i/p}$ and where $\mathbf{Z}$ denotes the integers. Because $w^j = w^{j \bmod p}$, the set W_p forms a subring of the complex numbers (i.e., W_p is closed under addition, multiplication, and negation). We use V_p to denote the set of all real members of W_p; that is, $V_p = W_p \cap \mathbf{R}$.

We now develop some properties of W_p that will prove useful in the analysis of linear programming to follow. Our immediate goal is to establish upper and lower bounds on certain functions of $\alpha \in W_p$. To obtain these bounds, we need a measure of the magnitude of the coefficients $a_0, \ldots, a_{p-1}$ in the representation $\alpha = \sum_{j=0}^{p-1} a_j\omega^j$. This representation, however, is not unique for general p, since $\sum_{j=0}^{p-1}\omega^j = 0$ for $p > 1$. The question of uniqueness of representation motivates us to measure the magnitude of the representation of α in the following manner:

Definition. Given $\alpha \in W_p$, we define the *representation height* of α with respect to W_p to be:

$$S_p(\alpha) = \min \left\{ \sum_{j=0}^{p-1} |a_j| : \alpha = \sum_{j=0}^{p-1} a_j\omega^j ; a_j \in \mathbf{Z}\ \forall j \right\},$$

where $\omega = e^{2\pi i/p}$. We include the index p on $S_p(\alpha)$ only when there is danger of confusion regarding the set W_p with respect to which the representation height of α is to be taken. (Representation height should not be confused with any of the various notions of height encountered in number theory.)

Next, we state the main algebraic and metric properties of the function S.

Proposition 3.1 *Let $\alpha, \beta \in W_p$. Then,*

(i) $S(a\alpha + b\beta) \leq |a|S(\alpha) + |b|S(\beta)$ *for any integers a and b,*

(ii) $S(\alpha\beta) \leq S(\alpha)S(\beta)$,

(iii) $|\alpha| \leq S(\alpha)$,

(iv) *If* $\alpha \neq 0$, $|\alpha| \geq (S(\alpha))^{1-p}$.

Proof. (Outline) Statements (i), (ii), and (iii) follow easily from the definition of representation height. To see that (iv) holds, consider $a \in \mathbf{Z}^p$ such that $S(\alpha) = \|a\|_1$, and let $A = \text{circ}\ (a)$. Then $|\alpha|^2$ is a nonzero eigenvalue of AA^T. Note that each eigenvalue of AA^T is bounded in magnitude by $S(\alpha)$. But, since AA^T is integer, the product of its nonzero eigenvalues has magnitude at least one. □

We shall use proposition 3.1 to derive several useful results concerning matrices and systems of equations whose coefficients belong to W_p. These results are most easily stated in terms of the quantities introduced below.

Definition. Let M be an $r\times s$ matrix all of whose entries M_{jk} belong to W_p. We define the *representation height of the matrix M* to be:

$$T(M) = \max_{j,k}\{S(M_{jk})\}.$$

Now, let $k = \max\{r, s\}$ and

$$\Delta(M) = (kT(M))^k.$$

Then, we define the *representation size of the matrix* M to be

$$L(M) = \log(\Delta(M)).$$

We use similar notation when discussing linear programs. Given $M \in \mathbf{R}^{r\times s}$, $d \in \mathbf{R}^r$, and $g \in \mathbf{R}^s$, we define $\Delta(M, d, g)$ and $L(M, d, g)$ to be $\Delta(Q)$ and $L(Q)$, respectively, where

$$Q = \begin{bmatrix} g^T & 0 \\ M & d \end{bmatrix}.$$

We refer to $L(M, d, g)$ as the *representation size of the linear program* $\{\min g^T v \,:\, Mv = d, v \geq 0\}$. We use analogous notation for the representation size of a system of linear equations.

Having fixed notation, we now state some key properties of matrix determinants.

Proposition 3.2 *Let B be a $r{\times}r$ nonsingular matrix all of whose entries B_{jk} belong to W_p, and let $\Delta = \Delta(B)$. Then,*

(i) $\det(B) \in W_p$,

(ii) $S(\det(B)) \leq \Delta$,

(iii) $\Delta^{1-p} \leq |\det(B)| \leq \Delta$.

As a corollary, we have the following result:

Corollary 3.1 *Let M and d be $r{\times}s$ and $r{\times}1$ matrices, respectively, all of whose entries belong to W_p, and let $\Delta = \Delta(M, d)$. If $\bar{v}$ is a basic solution to the system $Mv = d$, then every nonzero component $\bar{v}_j$ of $\bar{v}$ satisfies $\Delta^{-p} \leq |\bar{v}_j| \leq \Delta^p$.*

Using the above results, we can modify almost any variant of the interior point method [6] or the ellipsoid method [7] to solve problem (P) in polynomial time. In the remaining part of this section, we shall show how to modify the primal-dual path following algorithm and its analysis as presented in [11] and [12]. Since most of the algorithm and analysis are not affected by the change from rationals to V_p, we present only the necessary modifications.

For the purposes of the complexity analysis, we assume that we have a machine that performs addition, subtraction, multiplication, division, and comparison of real numbers in constant time per operation. Implicit in our derivation of the basic complexity results is the assumption that, for any instance of (P), a bound on the representation size of the instance is part of the input data. (For our purposes, this assumption is essentially equivalent to the requirement that $S(\alpha)$ be part of the input for every coefficient α in the problem instance.) Later, we adopt a more natural input scheme and show that the basic complexity results extend easily to this case.

Next, we use the properties of the set W_p and the function S to establish a theoretical stopping point for an interior point algorithm applied to (P) and (D). Because the discussion involves systems of inequalities and linear programs, we shall assume that the matrices and vectors we encounter are real.

Proposition 3.3 *Let the primal-dual pair (P) and (D) be given and let all the entries in A, b, and c belong to V_p. Assume we have a point $\bar{w} = (\bar{x}, \bar{y}, \bar{z})$ feasible to (P) and (D) that satisfies $\bar{x}^T\bar{z} \leq (6n\Delta)^{-24p}$, where $\Delta = \Delta(A, b, c)$. Then from $\bar{w}$, we can find a point $w^* = (x^*, y^*, z^*)$ in no more than $O(n^3)$ arithmetic operations, such that x^* is optimal to (P) and y^* and z^* are optimal to (D).*

Next, we state the minimum improvement made in the duality gap during each iteration of the primal-dual path following algorithm.

Proposition 3.4 *Let $w^0 = (x^0, y^0, z^0)$ be a valid initial point for the primal-dual path following algorithm given in [11] when applied to (P) and (D). Then the algorithm generates a sequence of feasible points $w^k = (x^k, y^k, z^k)$ satisfying*

$$(x^k)^T z^k \leq (x^0)^T z^0 (1 - \sqrt{1/n})^k.$$

Proof of proposition 3.4 follows directly from the fact that the convergence proofs given for the algorithm in [11] do not rely on the rationality of the input data.

We can now state the main results of the section.

Theorem 3.1 *Let $\delta \geq L(A, b, c)$ be given. Then, under the model of computation discussed above, problems (P) and (D) can be solved in time polynomial in p, n, and δ.*

Proof. For rational data, it is shown in [12] that the solutions to (P) and (D) can be obtained by solving a pair of artificial problems whose size is order of the size of (P) and (D). The artificial pair has the property that a valid starting point, with known duality gap, is readily available for the primal-dual algorithm. In our case, it is a straightforward exercise to show that we can construct a similar pair of artificial problems whose representation size is order of $L(A, b, c)$ and whose starting point has a duality gap that is $2^{O(p\delta)}$. Using propositions 3.3 and 3.4, it is easy to show that, given this initial duality gap, the number of iterations performed by the primal-dual algorithm is polynomial in p, n, and δ. Proof of the theorem then follows by noting that the work performed in each iteration of the algorithm is polynomial in n. □

Theorem 3.1 is derived under the assumption that a bound on the representation size of (P) is input along with the problem coefficients. As an alternative, we may consider a model of input based on an integer representation of the problem coefficients. In this model, we assume that every number $\alpha \in V_p$ in an instance of (P) is encoded for input as a set of integers $a_0, \ldots, a_{p-1}$, such that $\alpha = \sum_{j=0}^{p-1} a_j \omega^j$. To ensure α can be calculated from its integer representation, we also assume that the machine has available the real part of ω ($= \cos 2\pi/p$), or that it can calculate this number.

We work with a different measure of input size in the integer-based model. We define the *encoding size* of $\alpha \in W_p$ to be the sum of the binary encoding sizes of the integers $a_0, \ldots, a_{p-1}$ in the above expansion. Similarly, we define the encoding size of a matrix or LP to be the sum of encoding sizes of its coefficients.

We now restate our earlier complexity bounds in terms of the integer-based input model.

Theorem 3.2 *Under the model of computation discussed above, problems (P) and (D) can be solved in time polynomial in p, n, and the encoding size of (P).*

Proof. The result follows immediately from theorem 3.1 by noting that the encoding size of (P) is an upper bound on the representation size of (P). □

4 Tardos scheme for LP over a subring of the algebraic integers

In this section, we use the results of section 3 to modify the Tardos scheme for solving combinatorial linear programs [15]. The modifications permit us to solve problem (P) of section 3 in time polynomial in n, p, and the size of the matrix A, independent of the objective and right-hand side data. (In fact, the objective and right-hand side coefficients may be arbitrary real numbers.)

We shall follow the presentation of Tardos' algorithm given by Schrijver in [14, section 15.2], although we present only those key propositions needed for the switch from rationals to V_p.

We consider (P) and (D), the primal-dual pair of LPs defined at the beginning of section 3. In order to show that these problems can be solved in polynomial time independent of the numbers in b and c, we need several sensitivity results and a guarantee that we can find a feasible solution in time independent of the size of the right-hand side. We begin with the sensitivity results (propositions 4.1 and 4.2).

Let z' be defined as the solution of the following problem:

$$\begin{array}{lll} \min & \|z\|_2 & \\ \text{s.t.} & A^T y + z & = c. \end{array}$$

As in [14], we may assume $z' \neq 0$ without loss of generality. Now define the vector c' as $c' = \frac{mn\Delta^{p+1}}{\|z'\|_\infty} z'$. Note that c' can replace c in (P) without changing the optimal solution. Hence, we lose no generality by assuming that the objective vector c in (P) already has the form of c'.

Now, let $\tilde{c} = (\lceil c_1 \rceil, \ldots, \lceil c_n \rceil)^T$, where $\lceil c_k \rceil$ denotes the smallest integer not less than c_k, and consider the following primal-dual pair of LPs:

$$\begin{array}{llll} (P') & \min & \tilde{c}^T x & \\ & \text{s.t.} & Ax & = b \\ & & x & \geq 0 \end{array} \qquad \begin{array}{llll} (D') & \max & b^T y & \\ & \text{s.t.} & A^T y + z & = \tilde{c} \\ & & z & \geq 0. \end{array}$$

Using the properties of representation height and following the proofs given for rational problems by Schrijver [14], one can prove the following sensitivity results.

Proposition 4.1 *Let $(\tilde{y}, \tilde{z})$ be an optimal solution to (D') and let $\Delta = \Delta(A, b, \tilde{c})$. Then, $\|\tilde{z}\|_\infty \geq m\Delta^{p+1}$.*

Proposition 4.2 *Let $(\tilde{y}, \tilde{z})$ be an optimal solution to (D'), and suppose that (D) has an optimal solution. Let $\Delta = \Delta(A, b, \tilde{c})$. Then there exists an optimal solution (y^*, z^*) to (D) such that*

(i) $\|y^* - \tilde{y}\|_\infty \leq m\Delta^p$,

(ii) $\|z^* - \tilde{z}\|_\infty < m\Delta^{p+1}$,

(iii) $z_k^* > 0$, *where* $k = \arg\max_j \{\tilde{z}_j\}$.

Next, we state a result that is key in proving that a feasible solution to a linear program can be found in time independent of the size of the right-hand side (cf. [14, Lemma B]).

Proposition 4.3 *Let A be a $m \times n$ matrix of rank m, all of whose entries belong to V_p. Let $d = ((\Delta^p + 1), (\Delta^p + 1)^2, \ldots, (\Delta^p + 1)^n)^T$, where $\Delta = \Delta(A)$. Then every basic solution of the system $Ax = Ad$ is nondegenerate (i.e., for every $m \times m$ nonsingular submatrix B of A, the vector $B^{-1}Ad$ has no zero components).*

By following the Tardos scheme as presented in [14, section 15.2], one can easily (though tediously) verify that propositions 4.1 through 4.3 contain all the modifications to the proofs of Tardos' algorithm necessary for the switch from rationals to V_p.

Theorem 4.1 *Let $\delta \geq L(A)$ be given. Then problems (P) and (D) can be solved in time polynomial in p, n, and δ.*

Proof. Note that the effort involved in rounding off the objective and right-hand side vectors in Tardos' algorithm depends only on the size of m, n, and $\Delta(A)$ and not on the size of b or c, since these vectors are scaled before rounding. Therefore, the rounding procedure is polynomial even if b and c are real.

Because we have available the polynomial-time algorithm for LPs with coefficients in V_p developed in section 3, the result follows by the arguments in [14, section 15.2] together with propositions 4.1 through 4.3. □

The next theorem follows immediately from theorem 4.1 by noting that the encoding size of A is an upper bound on $L(A)$.

Theorem 4.2 *Problems (P) and (D) can be solved in time polynomial in p, n, and the encoding size of the matrix A.*

We are now able to prove the claim, made in section 2, that LPs with real circulant coefficient matrices are strongly polynomial.

Proof of Theorem 2.2. By the discussion in section 2, we have an apriori bound on the representation size of the coefficient matrix, namely $L(nA^+A) \leq 2n \log n$. Proof of the theorem follows directly from this bound and theorem 4.1. □

5 LP in quadratic field extensions

In this section, we use our earlier results to obtain the complexity of linear programs in which the coefficients are integer linear combinations of integer square roots. In particular, we consider LPs whose coefficients belong to $\mathbf{Z}(d_1, \ldots, d_k)$, where we define $\mathbf{Z}(d_1, \ldots, d_k)$ to be the additive and multiplicative ring generated by $1, \sqrt{d_1}, \ldots, \sqrt{d_k}$; that is, $\mathbf{Z}(d_1, \ldots, d_k)$ consists of all numbers that have the form $\sum a_j (\sqrt{d_1})^{j_1} (\sqrt{d_2})^{j_2} \cdots (\sqrt{d_k})^{j_k}$, where a_j is an integer and the summation runs over all (distinct) k-tuples $J_j = (j_1, \ldots, j_k)$ with elements that are either 0 or 1. Because we are interested in linear programs with real coefficients, we shall assume that the d_j are positive.

Our strategy is to find an integer q such that the set $\mathbf{Z}(d_1, \ldots, d_k)$ is embedded in the set W_q. Using this result, we bound the representation height of $\alpha \in \mathbf{Z}(d_1, \ldots, d_k)$ with respect to W_q by a function of the coefficients in the representation of α in terms of the cross products of the $\sqrt{d_j}$. We then apply the results of sections 3 and 4, which bound the complexity of a LP by a function of its representation size.

We begin by stating a key result due originally to Gauss. Proof can be found in many advanced texts on number theory (see, e.g., [5]).

Proposition 5.1 *Let p be an odd prime and let $\omega = e^{2\pi i/p}$. Then,*

$$\sum_{j=0}^{p-1} \omega^{j^2} = \begin{cases} \sqrt{p} & \text{if } p = 1 \bmod 4 \\ i\sqrt{p} & \text{if } p = 3 \bmod 4. \end{cases}$$

Note that, for any k, the set W_{4k} contains both W_k and $i = \sqrt{-1}$. It follows that if p is a prime and $p = 3 \bmod 4$, then W_{4p} contains $\mathbf{Z}(p) = \{\alpha : \alpha = a + b\sqrt{p}; a, b \in \mathbf{Z}\}$. To complete the characterization of prime numbers, we make the easily verified observation that W_8 contains $\mathbf{Z}(2)$.

Using the above results and the prime factorization theorem for integers, we can characterize an embedding of $\mathbf{Z}(d)$ for d not necessarily prime:

Proposition 5.2 *Let d be a positive integer. Then W_{4d} contains $\mathbf{Z}(d)$. Moreover, $S_{4d}(\sqrt{d}) \leq d$.*

Making an easy generalization of proposition 5.2, we next characterize an embedding of the ring generated by a number of square roots.

Proposition 5.3 *Let $d_1, \ldots, d_k$ be positive integers and let $d = \prod_{j=1}^{k} d_j$. Then, W_{4d} contains $\mathbf{Z}(d_1, \ldots, d_k)$.*

Proposition 5.4 *Let $d_1, \ldots, d_k$ be positive integers and let $d = \prod_{j=1}^{k} d_j$. Let $\alpha \in \mathbf{Z}(d_1, \ldots, d_k)$ have the representation $\alpha = \sum a_j (\sqrt{d_1})^{j_1} (\sqrt{d_2})^{j_2} \cdots (\sqrt{d_k})^{j_k}$, where a_j is an integer and the summation runs over all (distinct) k-tuples $J_j = (j_1, \ldots, j_k)$ with elements that are either 0 or 1. Then, $S_{4d}(\alpha) \leq d \sum |a_j|$.*

We now apply these results to bound the complexity of LPs whose coefficients belong to $\mathbf{Z}(d_1, \ldots, d_k)$. As in sections 3 and 4, we assume that we have a machine that can perform arithmetic operations on real numbers in constant time per operation, and that the machine has available $\sqrt{d_j}$, for all j, or that it can calculate these numbers. We also assume that every number $\alpha \in \mathbf{Z}(d_1, \ldots, d_k)$ in a problem instance is encoded for input as a set of 2^k integers that are the coefficients in the representation of α in terms of the cross products of the $\sqrt{d_j}$. We define the *encoding size* of α to be the sum of the binary encoding sizes of these coefficients, and we define the encoding size of a matrix to be the sum of the encoding sizes of its entries.

Theorem 5.1 *Let (P) be the standard form LP defined at the beginning of section 3. Suppose that all the coefficients in (P) belong to $\mathbf{Z}(d_1, \ldots, d_k)$ for positive integers $d_1, \ldots, d_k$ with $d = \prod_{j=1}^{k} d_j$. Then (P) can be solved in time polynomial in n, d, and the encoding size of the matrix A.*

Proof. Using proposition 5.4, it is easy to show that the encoding size of A is an upper bound on $L(A)$, the representation size of A. Proof of the theorem then follows from theorem 4.1. □

If d is fixed, theorem 5.1 implies that (P) can be solved in time polynomial in the problem dimension and encoding size. We improve these results in a sequel paper [1], obtaining a bound which, although exponential in k, is polynomial in the bit size of d and the problem encoding size.

6 Remarks

In light of the results obtained here on the complexity of LPs with coefficients from W_p, it may be worth investigating what interesting classes of real numbers can be embedded in W_p. It is well-known that every finite Abelian extension of the rationals can be embedded in the extension of the rationals by the p^{th} root of unity, for some p (see, e.g., [5]). Therefore, there may be other classes of LPs whose complexity can be bounded in the manner developed in section 5 for LPs with coefficients from a square-root extension of the integers.

More ambitiously, one may ask if it is possible to obtain complexity results for other classes of algebraic numbers without first embedding the numbers in W_p. Recently, by using some additional material from number theory in conjunction with an approach similar to that of sections 3 and 4, we have obtained such results for all algebraic numbers [1].

In some sense, our approach in this paper has been to encode a set of algebraic numbers as integers for input to a machine that performs real arithmetic. It is natural to ask whether the requirement for real arithmetic can be relaxed to the point where all computations are performed symbolically, using integer arithmetic only. We believe this can be done both here and in the more general context of an algorithm for all algebraic numbers, and we plan a subsequent report on this question.

Acknowledgement

This research was funded by the National Science Foundation under grant DMS8810192. We greatly appreciate this financial support.

References

[1] I. Adler and P. A. Beling, "Polynomial Algorithms for Linear Programming over the Algebraic Numbers," manuscript, July, 1991.

[2] L. Blum, M. Shub and S. Smale, "On a Theory of Computation and Complexity over the Real Numbers; NP-completeness, Recursive Functions and Universal Machines," *Bulletin of the AMS* 21, No. 1, pp. 1-46, 1989.

[3] P. J. Davis, *Circulant Matrices*, John Wiley and Sons, New York, 1979.

[4] J. Edmonds, "System of Distinct Representatives and Linear Algebra," *J. Res. Nat. Bur. Standards*, **71/B**, pp. 241-245, 1967.

[5] K. Ireland and M. Rosen, *A Classical Introduction to Modern Number Theory*, Springer-Verlag, New York, 1972.

[6] N. Karmarkar, "A New Polynomial Time Algorithm for Linear Programming," *Combinatorica* **4**, pp. 373-395, 1984.

[7] L. Khachiyan, "A Polynomial Algorithm in Linear Programming," *Soviet Mathematics Doklady* **20**, pp. 191-194, 1979.

[8] N. Megiddo, "Towards a Genuinely Polynomial Algorithm for Linear Programming," *SIAM Journal on Computing* **12**, pp. 347-353, 1983.

[9] N. Megiddo, "Linear Programming in Linear Time When the Dimension is Fixed," *Journal of the Association for Computing Machinery* **31**, pp. 114- 127, 1984.

[10] N. Megiddo, "On Solving the Linear Programming Problem Approximately," *Contemporary Mathematics*, 1990.

[11] R. D. C. Monteiro and I. Adler, "Interior Path Following Primal-dual Algorithms. Part I: Linear Programming," *Mathematical Programming* **44**, pp. 27-41, 1989.

[12] R. D. C. Monteiro and I. Adler, "Interior Path Following Primal-dual Algorithms. Part II: Convex Quadratic Programming," *Mathematical Programming* **44**, pp. 43-66, 1989.

[13] C. Norton, S. Plotkin and E. Tardos, "Using Separation Algorithms in Fixed Dimension," *Proceedings of the 1st ACM/SIAM Symposium on Discrete Algorithms*, pp. 377-387, 1990.

[14] A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley and Sons, New York, 1987.

[15] E. Tardos, "A Strongly Polynomial Algorithm to Solve Combinatorial Linear Programs," *Operations Research* **34**, pp. 250-256, 1986.