AN ALGORITHM FOR FINDING A SHORTEST VECTOR IN A
TWO DIMENSIONAL MODULAR LATTICE

by

Mody Lempel and Azaria Paz

Technical Report #658

December 1990
AN ALGORITHM FOR FINDING A SHORTEST VECTOR IN A TWO DIMENSIONAL MODULAR LATTICE

by

Mody Lempel and Azaria Paz
Computer Science Department
Technion - Israel Institute of Technology
Haifa 32000, Israel

ABSTRACT

Let \( 0 < a, b < d \) be integers with \( a \neq b \). The lattice \( L_d(a, b) \) is the set of all multiples of the vector \( (a, b) \) modulo \( d \). An algorithm is presented for finding a shortest vector in \( L_d(a, b) \). The complexity of the algorithm is shown to be logarithmic in the size of \( d \), when the number of arithmetical operations are counted.
1. INTRODUCTION

A classical algorithm, due to Gauss, for finding a shortest vector in a two dimensional lattice, has been used as one of the main building blocks in the recent $L^3$ (Lenstra A.K., Lenstra, H.W. Jr., and Lovasz, L.) basis reduction algorithm for general lattices [1]. The complexity of Gauss' algorithm has been shown to be logarithmic in the maximal integer among the entries of the vectors forming the basis of the lattice at input (when counting the number of arithmetical operation involved) [2].

Let $0 < a, b < d$ be integers such that $a \neq b$. We define $L_d(a,b)$ to be the modular lattice generated by the vector $(a,b)$ modulo $d$, i.e. the (finite) set of all vectors of the form $(ia \pmod{d}, ib \pmod{d})$, $0 \leq i < d$, which is closed under addition modulo $d$.

We present, in this paper, an algorithm for finding a shortest vector in a lattice $L_d(a,b)$ as above, and we prove that the complexity of the algorithm is logarithmic in the size of $d$, when counting the number of arithmetical operations involved.

While our algorithm bears certain similarities to the algorithm of Gauss, the two algorithms are different and cannot be reduced one to the other, when Gauss algorithm is considered over vectors with integer entries only. Thus, e.g. in the modular lattice generated by the vector $(4,1)$ modulo 5, a shortest (nonzero) vector is the vector $(2,3) = (3-4, 2-1) \pmod{5}$, i.e. the (finite) set of all vectors of the form $(ia \pmod{5}, ib \pmod{5})$, $0 \leq i < 5$, which is closed under addition modulo 5.

Conversely, consider the general (nonmodular) lattice with base vectors $(7,11)$ and $(5,8)$. The determinant $\begin{vmatrix} 7 & 11 \\ 5 & 8 \end{vmatrix}$ is equal to 1. It can be shown that under those circumstances no $d > 1$ exists such that $(5,8) = (i 7 \pmod{d}, i 11 \pmod{d})$, $0 \leq i < d$, since the existence of such a $d$ would imply that the above determinant has value $\geq d$ (see next section in the paper).

It is hoped that this algorithm will enable generalizations for general $n$-dimensional modular lattices, and will have applications to other areas of study (e.g. Cryptology, Coding Theory, Geometry of Numbers, etc.).
2. PRELIMINARIES

Given the integers 0 < a, b < d and i, the notation i (a, b)(mod d) stands for the vector (ia (mod d), ib (mod d)).

We shall denote by L_d(a, b) the modular lattice L_d(a,b) = \{ (a,b)(mod d): 0 \leq i \leq d-1 \}. We start with a few simple remarks:

(1) If \gcd(a,b,d) = g > 1 then the lattice L_d[g \left( \frac{a}{g}, \frac{b}{g} \right)] is an isomorphic contraction of the lattice L_d(a,b). The shortest vector of the original lattice is equal to the shortest vector of the L_d[g] lattice multiplied by g. We shall assume therefore that \gcd(a,b) = 1.

(2) If \gcd(a,b,d) = 1 but \gcd(a,b) = g_1 > 1 then L_d \left( \frac{a}{g_1}, \frac{b}{g_1} \right) = L_d(a,b). This follows from the fact that g_1 is invertible modulo d, given that \gcd(a,b,d) = 1. Thus

i (a,b)(mod d) = \left[ \frac{a}{g}, \frac{b}{g} \right] (mod d)

where

j = ig (mod d) if i is given and
i = jg^{-1} (mod d) if j is given.

(3) Any two vectors in L_d whose determinant is equal to \pm d will be called a geometrical basis for L_d. It will be shown in the next section that any vector (a,b) = (1,1) (with \gcd(a,b) = 1) belongs to a geometrical basis. Vectors in L_d will be considered both as vectors and as points in two dimensional space. Let (a,b) and (c,e) be a geometrical basis in L_d. Consider the topological torus formed from the square 0 \leq x, y \leq d when its edges x = 0, y = 0 are identified with the edges x = d, y = d, correspondingly. The area of the face of this torus is d^2 and it is covered by d nonoverlapping translates of the parallelogram whose vertices are (0,0),(a,b),(c,e),(a+c,b+e) and whose area is d. It follows that the determinant of any 2 points in L_d which are not colinear is equal to \pm kd where k is an integer 0 < k < d.
3. SOME PROPERTIES OF $L_d$

Lemma 1: Let $(c, e)$ be a point vector in $L_d(a, b)$ such that $gcd(c, e) = 1$. If $c > 1$ then there is a vector $(c_1, e_1)$ in $L_d(a, b)$ such that $\begin{vmatrix} c & e \\ c_1 & e_1 \end{vmatrix} = d$. If $e > 1$ then there is a vector $(c_2, e_2)$ in $L_d(a, b)$ such that $\begin{vmatrix} c & e \\ c_2 & e_2 \end{vmatrix} = -d$.

Proof: Assume that $c > 1$. $gcd(c, e) = 1$ implies that there are integers $u, v$ such that $cu - ev = 1$. Multiplying by $d$ we get $cud - evd = d$. This equality induces the set of equalities $c(ud - ke) - e(vd - kc) = d$ for any integer $k$.

Let $k_0$ be the maximal $k$ such that both $(ud - k_0e) = e_1 = 0$ and $(vd - k_0c) = c_1 = 0$. If both $e_1$ and $c_1$ are smaller than $d$, then from $(c_1, e_1) = (v, u)d - (c, e)k_0$ we get that $(c_1, e_1)$ is a modular multiple of a vector in $L_d(a, b)$ and satisfies therefore the requirement of the lemma. To complete the proof of the first part of the lemma we must show that $c_1, e_1 < d$.

From the choice of $k_0$ we know that either $c_1 < c$ or $e_1 < e$. Assume, by way of contradiction that $ce_1 - ec_1 = d$ and either $c_1 < c < d$ together with $e_1 \geq d$ or $c_1 \geq d$ together with $e_1 < e < d$.

In the first case we have that $c_1 \leq c - 1, c < d$ and $e_1 \geq d$. Also, since $c > 1$ (by assumption) $c-1 > 0$. Therefore, $d = ce_1 - ec_1 > cd - d(c-1) = d$, a contradiction.

In the second case, we have that $e_1 \leq e - 1, c < d$ and $c_1 \geq d$. This implies that $d = ce_1 - ec_1 < d(e - 1) - ed = -d$, which is impossible.
It follows that \( c_1, e_1 < d \) and the proof of the first part of the lemma is complete. The proof of the second part is similar.

Remark: The excluded point vector \((1,1)\) can never belong to a geometrical basis since the value of the determinant \[
\begin{vmatrix}
1 & 1 \\
c & e
\end{vmatrix}
\] is always less than \(d\), in absolute value, given that \(0 \leq c, e < d\). Moreover, if the vector \((1,1)\) belongs to a modular lattice \(L_d\), then

\[
L_d = \{(k,k); 0 \leq k < d\}.
\]

No other vector \((c,e) \neq (k,k)\) can belong to \(L_d\). Any such vector forms a determinant with \((1,1) \in L_d\) whose value is less than \(d\), which cannot happen for vectors in \(L_d\) (see previous section).

The next few lemmas provide a characterization of the set of points forming a geometrical basis with a given vector.

Lemma 2: Let \((a,b)\) be a vector in a lattice \(L_d\). Let \((c,e)\) be another vector in \(L_d\) such that \((a,b)\) and \((c,e)\) form a geometrical basis. If \(gcd(a,b) = g > 1\) with \((a,b) = g(a',b')\) then any vector of the form \(k(a',b')\), \(1 \leq k < g\) is not in \(L_d\).

Proof: Since \((a,b)\) and \((c,e)\) form a basis we have that \[
\begin{vmatrix}
a & b \\
c & e
\end{vmatrix} = \begin{vmatrix} ga' & gb' \\
c & e
\end{vmatrix} = \pm d.
\]
Assume that the determinant equals \(+d\) (the other case is similar). This implies that

\[
0 < \left| \begin{array}{cc}
ka' & kb' \\
c & e
\end{array} \right| < \left| \begin{array}{cc}
 ga' & gb' \\
c & e
\end{array} \right| = d.
\]

Given that \((c,e)\) is in \(L_d\), if \(k(a',b')\) is in \(L_d\) then \[
\left| \begin{array}{cc}
 ka' & kb' \\
c & e
\end{array} \right|
\]

must be equal to 0 or to a nonzero multiple of \(d\), a contradiction.

Lemma 3: Let \((a,b)\) be a vector in a lattice \(L_d\) and let \((c,e)\) and \((c',e')\) be two vectors in \(L_d\) such that both form a basis with \((a,b)\). Then \((c,e)\) can be written in the form

\[
(c,e) = (c',e') + i(a,b) \quad \text{or} \quad (c,e) = -(c',e') + i(a,b)
\]

for some \(1 \leq i < d\).

Proof: It follows from the assumptions that \(ae - bc = \pm (a'e - b'c')\). Let \(gcd(a,b) = g\) with \((a,b) = g(a',b')\). Then

\[
\begin{align*}
\begin{vmatrix} ga' & gb' \\
c & e
\end{vmatrix} &= gb'(c \pm c') \\
\end{align*}
\]

implying that

\[
e \pm e' = kb' \quad \text{and} \quad c \pm c' = ka'
\]

- 5 -

Technion - Computer Science Department - Technical Report CS0658 - 1990
Thus

\[(c,e) = (c',e') + k(a',b') \text{ or } (c,e) = -(c',e') + k(a',b').\]

If \(g = 1\) or \(g \mid k\) then we are done. To complete the proof we show that this is the only possible case.

Otherwise, let \(g > 1\) and \(k = gs + r, 0 < r < g\). Then

\[(c,e) = (c',e') + s(a,b) + r(a',b')\]

or

\[(c,e) = -(c',e') + s(a,b) + r(a',b').\]

In both cases \(r(a',b')\) must be in \(L_d\) since \((c,e), (c',e')\) and \(s(a,b)\) are in \(L_d\) and all the entries of all the vectors involved are nonnegative. But this contradicts Lemma 2 since \(0 < r < g\).

\[\square\]

**Corollary 4:** Let \((a,b), (c,e)\) be two vectors in \(L_d\) which are a geometric basis. The set of all vectors forming a basis with \((a,b)\) in \(L_d\) is the set (*) below

\[
\{(p,q) = \pm (c,e) + i(a,b): -d \leq i \leq d, 0 \leq p,q < d\}
\]

(*)

\[\]

**Lemma 5:** Let \(i_0\) be the maximal \(i\) such that \((c,e) - i_0(a,b)\) is nonnegative and let \(i_1\) be the minimal \(i\) such that \(-(c,e) + i_1(a,b)\) is nonnegative, in the set (*). Then the shortest vector in the set (*) is the shortest of \((p',q')\) and \((p'',q'')\) where

\[(p',q') = (c,e) - i_0(a,b),\]

\[(p'',q'') = -(c,e) + i_1(a,b).\]

**Proof:** Left to the reader.

**Remark:** Notice that \(i_0\) can be defined as

\[
i_0 = \begin{cases} \lfloor \frac{c}{a} \rfloor & \text{if } a,b > 0 \text{ then } \min(\lfloor \frac{c}{a} \rfloor, \lfloor \frac{c}{b} \rfloor) \\ \lfloor \frac{c}{b} \rfloor & \text{if } b = 0 \text{ then } \lfloor \frac{c}{a} \rfloor \\ \lfloor \frac{c}{a} \rfloor & \text{if } a = 0 \text{ then } \lfloor \frac{c}{b} \rfloor \end{cases}
\]

and \(i_1\) can be defined in a similar way. It follows that the number of operations involved in the computation of the shortest vector in the set (*) is constant.
2.4 If \( p \leq m - 1 \) return \( u_p \), halt;

2.5 Set \( v_2 = u_p \), \( v_1 = u_{p-1} \), \( k := \left\lfloor \frac{v_1}{v_2} \right\rfloor \)

end;  
(\text{Remark: Now } k > 2 \text{ and } \delta_j > 0) \)

2.6 Set \( v_3 := v_2 \), \( v_2 := -v_1 + kv_2 \), \( v_1 := v_3 \)

end (repeat);

3. If \( |v_2| > |v_1| \) return \( v_1 \), else return \( v_2 \)

end of algorithm

5. PROPERTIES OF THE MIN-CROSS PROCEDURE

The vectors at input \( v_1 \) and \( v_2 \) are assumed to satisfy the following properties:

(a) \( |v_2| < |v_1| \)
(b) \( v_2 \) and \( v_1 \) are cross vectors
(c) \( v_2 \) and \( v_1 \) are a geometrical basis for a lattice \( L_d \).

Consider the sequence of vectors below:

\[ v_1, v_2, v_3, \ldots, v_i \]

such that for all \( i > 2 \) the following properties hold

(d) \( |v_i| < |v_{i-1}| \), \( i \geq 2 \)
(e) \( v_i \) is the shortest vector in \( L_d \) which forms a basis for \( L_d \) with \( v_{i-1} \).

We proceed to prove the following

Theorem 6:

(f) \( v_i \) and \( v_{i-1} \) are cross vectors, of the same type (left or right) as \( v_2 \) and \( v_1 \), for all \( i \geq 2 \).

(a) The vectors generated, at steps 2.5 and 2.6 by the procedure Min-Cross are a subsequence of the sequence (1), starting from \( v_2 \) and on.

(b) \( v_i \), the last vector in (1), is the last vector generated by the procedure, at one of the steps 1, 2.3, 2.4, 3.

(i) The number of iterations of the procedure is logarithmic in the magnitude of \( d \).

Proof: (f) is proved by induction. By assumption \( v_2 \) and \( v_1 \) are cross vectors. Assume that \( v_{i-1} \) and \( v_{i-2} \), \( i \geq 3 \), are left cross vectors with \( v_{i-1,1} < v_{i-2,1} \) and \( v_{i-1,2} > v_{i-2,2} \) (the right cross case is
where the line passing through \( v_1 \) and \( v_2 \) is
\[
I(\alpha) = v_2 - \alpha(v_1 - v_2) = v_2 - \alpha z.
\]

Let \( m = \left\lfloor \frac{v_{21}}{\delta_1} \right\rfloor \) and assume \( k = \left\lfloor \frac{v_{11}}{v_{21}} \right\rfloor = 2.\)

**Claim 1:** \( k = 2 \) implies that \( m \geq 1.\)

**Proof:** \( k = 2 \) implies that \( v_{11} \leq 2v_{21} \) by the definition of \( k. \) This implies that \( \delta_1 = v_{11} - v_{21} \leq v_{21} \) or \( \frac{v_{21}}{\delta_1} \geq 1 \) resulting in \( m = \left\lfloor \frac{v_{21}}{\delta_1} \right\rfloor \geq 1. \)

Define as before \( u_1 = v_2 - i\delta \) (\( u_{-1} = v_1, \delta = v_1 - v_2 \)), \( k_i = \left\lfloor \frac{u_{i-1}}{u_{i}} \right\rfloor \) \((k = k_0)\), and assume the following: \( k_0 = 2, u_0 \) is the shortest vector in \( L_d \) forming a basis with \( u_{-1} \) and \( u_0, u_{-1} \) are left cross vectors.

**Claim 2:** Under the above assumptions, for all \( 0 \leq i < m, k_i = 2 \), the vector \( u_i \) is the shortest vector forming a basis in \( L_d \) with \( u_{i-1}, u_i \) and \( u_{i-1} \) are left cross, and \( u_{i-1} - u_i = \delta.\)

**Proof:** By induction. For \( i = 0 \), the properties follow from the definitions and assumptions.

Assume now that, for \( i > 0, k_{i-1} = 2, u_{i-1} \) is the shortest vector forming a basis in \( L_d \) with \( u_{i-2}. u_{i-1} \) and \( u_{i-2} \) are left cross vectors and \( u_{i-2} - u_{i-1} = \delta. \) The shortest vector forming a basis in \( L_d \) with
If $\alpha < 0$ (step 3.1, $p_1 = [\bar{\alpha}]$), then there is no vector shorter than $v_2$ forming a basis for $L_d$ with $v_1$ (the shortest such vector is the vector $v_2 - \delta$ which must be longer than $v_2$ since $v_2$ is longer than $v_2 + \langle -\alpha \delta \rangle$). $v_2$ must therefore be the final vector in the sequence (1) and the procedure halts.

Let $u_r$ be the shortest vector in the sequence ($u_i$) on $I$.

If $r = m$: this happens if $p_1 = [\bar{\alpha}] = m$ or $p_1 = [\bar{\alpha}] = m - 1$ but $p = m$ ($\|u_m\| < \|u_{m-1}\|$). Then the sequence $u_1, u_2, \ldots, u_m$ is the subsequence $v_1, v_2, \ldots, v_{m+1}$ of (1), by Claim 2 with $m \geq 1$ (Claim 1). This case corresponds to step 2.5 in the procedure. The vectors $v_1$ and $v_2$ are reset and the procedure continues with step 3.

If $r < m$, then $u_r$ is one of the vectors $v_2 - [\bar{\alpha}] \delta$, $v_2 - [\bar{\alpha}] \delta$, $u_{r+1}$ with $r + 1 \leq m$ is the shortest vector in $L_d$ which forms a base with $u_r$. But in this case $\|u_{r+1}\| > \|u_r\|$ and therefore the sequence (1) terminates with $u_r = v_{r+2}$. If this is the case, the procedure halts with $u_r$ at output.

If $k \geq 3$ then the procedure proceeds directly to step 2.6 and it either halts with $v_1$ at output, if the new $v_2$ (the shortest vector forming a basis with $v_1$) is longer than $v_1$; or it halts with the new $v_2$ at output, if the new $v_2$ is terminal; or it proceeds with a new iteration. The proof of the properties (g) and (h) is thus complete.

Proof of (i): Let $v_{j+2}$ be the new $v_2$ vector created at step 2.6 at iteration $j$. The application of step 2.6 is based on $k > 2$. Therefore $k = [\frac{v_{j+1}}{v_{j+2}}] > 2$ or $v_{j+1,1} > 2v_{j+2,1}$. Thus the new first coordinate of $v$ is decreased by a factor of at least 2. The number of iterations is therefore logarithmic in the magnitude of the coordinates of the vectors at input which are bounded by $d$.

All properties of the procedure are now proved.

6. THE MAIN ALGORITHM

To find the shortest vector in a modular lattice $L_d$ generated by a vector $v = (v_1, v_2)$, modulo $d$, $v_1 \neq v_2$, apply the following algorithm.

1. Assume $\gcd (a, b, d) = 1$
2. If $\gcd (v_1, v_2) = g > 1$ then reset $(v_1, v_2) = \frac{1}{g}(v_1, v_2)$. Now $(v_1, v_2) \neq (1, 1)$.
3. Based on Lemma 1 and Lemma 5 find the shortest vector $v_2$ forming a basis with $v_1$
4. While $\|v_2\| < \|v_1\|$
4.1 If \( v_2 \) and \( v_1 \) are crossing call \( \text{Min-Cross} (v_1, v_2) \).
4.2 \( v_1 := v_2 \).
4.3 Based on Lemma 5 find the shortest vector \( v_2 \) forming a basis with \( v_1 \).

5. Return \( v_1 \).

We conclude now by showing that the algorithm is correct and that its complexity is logarithmic in the size of \( d \) (when counting the number of arithmetical operations).

7. PROOF OF CORRECTNESS

When the algorithm terminates, either via the \( \text{mincross} \) procedure or at step 5, it produces a vector \( v \) such that no vector in \( L_d \) forming a basis with \( v \) is shorter than \( v \). We claim that such a vector \( v \) is the shortest vector in \( L_d \). We need first the following.

Lemma 7: Let \( ABC \) be a triangle in the plane such that the vertices \( A, B, C \) correspond to vectors in \( L_d \). If the area of \( ABC \) is greater than \( d/2 \) then there must be a point of \( L_d \) different from \( A, B, C \), on the border of or inside the triangle.

Proof: Consider the torus formed by identifying the edges \( x = d, y = 0 \) with the edges \( x = 0, y = 0 \), correspondingly, of the square \( \{(x,y) : 0 \leq x,y \leq d\} \). The area of the face of this torus is \( d^2 \). If no lattice point exists inside or on the border of \( ABC \) then the parallelogram formed by the edges \( AB \) and \( AC \) has no lattice points inside or on its border, except its vertices. Under the assumption of the lemma, the area of the parallelogram is greater than \( d \). Thus \( d \) translates of this parallelogram will cover the whole torus with no overlap implying that the area of the torus is greater than \( d^2 \), a contradiction.

Theorem 8: If a vector \( v \) in \( L_d \) has the property that no vector in \( L_d \) forming a basis with \( v \) is shorter than \( v \), then \( v \) is the shortest vector in \( L_d \).

Proof: Assume to the contrary that there is a vector \( v_1 \) shorter than \( v \) in \( L_d \). \( v_1 \) cannot form a basis with \( v \) by the properties of \( v_1 \). Therefore, the triangle whose vertices are \( 0, v, v_1 \) (\( 0 \) is the origin) must have an area which is greater than \( d/2 \). Both vectors \( v \) and \( v_1 \) belong to some basis and therefore, by Lemma 2, no vector in \( L_d \) can subdivide \( v_1 \) or \( v_2 \). By the previous lemma there must be a point in \( L_d \) inside the triangle or on the line joining \( v \) to \( v_1 \). Let \( v_2 \) be such a point, then obviously \( |v_2| < |v| \) (since \( |v_1| < |v_1| \)) and the area of the triangle whose vertices are \( 0, v_2, v \) is smaller than the area of
the original triangle. Choose \( v_2 \) to be a point as above and such that the area of the triangle whose vertices are \( 0, v_2, v \) is minimal. It follows from the choice of \( v_2 \) that no vector in \( L_d \) can be inside the minimal triangle or on its \((0,v_2)\) or \((v_1,v_2)\) boundaries. Now \( v_2 \) cannot form a basis with \( v \) since \(|v_2| < |v|\). Therefore, the area of the minimal triangle must be greater than \( d/2 \). But this contradicts, by the previous lemma, the fact that no points of \( L_d \) exist inside or on the boundary of this minimal triangle.

The algorithm is thus shown to be correct.

8. COMPLEXITY ANALYSIS

If, at step 4, \(|v_2| \geq |v_1|\), the algorithm halts. If at step 4.1 \( v_2 \) and \( v_1 \) are crossing, then the algorithm enters procedure Min-Cross and will eventually halt, while executing this procedure, in at most \( O(\log d) \) steps.

Let \( \mathbf{v}_i, \mathbf{v}_{i-1}, \mathbf{v}_{i-2} \) be the vectors generated at step 4.3 at the \( i, i-1 \) and \( i-2 \) iterations correspondingly, with \( i \geq 2 \). Since \(|v_{i-1}| < |v_{i-2}| \) with \(|v_{i-1}|^2 \) and \(|v_{i-2}|^2 \) integers, the algorithm will eventually halt. Since the algorithm did not enter the procedure mincross at step 4.1, we must have that \(|v_{i-1}| < |v_{i-2}| \) and \( v_{i-1}, v_{i-2} \) are not crossing. Therefore \( v_{i-1,1} \leq v_{i-2,1}, v_{i-1,2} \leq v_{i-2,2} \), and at least one of the inequalities is strict. Let

\[
k_i = \min \left( \frac{|v_{i-2,1}|}{|v_{i-1,1}|}, \frac{|v_{i-2,2}|}{|v_{i-1,2}|} \right); \quad k_i \geq 1 \quad \text{(since the vectors are not cross).}
\]

The vector \( \mathbf{v}_i \) generated at step 4.3 is either equal to \( v_{i-2} - k_i v_{i-1} \) or a shorter vector (in case \(-v_{i-2} + k_i v_{i-1}\), as defined in Lemma 5, is shorter than \( v_{i-2} + k_i v_{i-1} \)). Set \( \mathbf{v}'_i = v_{i-2} - k_i v_{i-1} \). It follows that

\[
|v'_i| = |v_{i-2} - k_i v_{i-1}| \geq |v_i|.
\]

Now \( v_{i-2} = v' + k_i v_{i-1} \) which implies that

\[
|v_{i-2}| = |v' + k_i v_{i-1}| \geq |v'_i + v_{i-1}|
\]

since \( k_i \) is positive and the entries of the vectors involved are nonnegative.

Consider the parallelogram whose vertices are the origing \( O \) and the points \( A, B, C \), corresponding to \( v_{i-1}, v'_i \) and \( v'_i + v_{i-1} \), all in the positive quadrant. Since \( v'_i \) and \( v_{i-1} \) are both in the positive quadrant, the origin is an acute angle in the parallelogram and the angle between the edges \( OA \) and \( AC \) is obtuse. It follows from the law of cosines that \( OC^2 \geq OA^2 + AC^2 = OA^2 + OB^2 \) which implies that
Combining the last three inequalities we get that

\[ |v_{i+1}|^2 \geq |v_i|^2 + |v_{i-1}|^2. \]

Notice also that the numbers involved in the above inequality are nonnegative integers.

Let \( t \) be the number of iterations of the algorithm through step 4.3 and let \( \phi \) be the positive solution of the equation \( x^2 = x+1 \), \( \phi = (1+\sqrt{5})/2 \). Then

\[ |v_1|^2 \geq 1 \]
\[ |v_{i-1}|^2 \geq |v_i|^2 \geq 1 \]
\[ |v_{i-2}|^2 \geq |v_{i-1}|^2 + |v_i|^2 \geq 2 > \phi \]
\[ |v_{i-3}|^2 \geq |v_{i-2}|^2 + |v_{i-1}|^2 > \phi + 1 = \phi^2 \]
\[ |v_{i-j}|^2 \geq |v_{i-j+1}|^2 + |v_{i-j+2}|^2 > \phi^{j-1} \]
\[ |v_0|^2 = |v_{t-1}|^2 > \phi^{t-1} \]

We get that

\[ (t-1)\log \phi < \log |v_0|^2 < 4\log d \]

or

\[ t < \frac{4\log d}{\log \phi} + 1 \]

The complexity of the algorithm is thus shown to be logarithmic in the magnitude of \( d \). If the algorithm does not enter the mincross procedure then the number of iterations before it halts is bounded as above. If it enters the procedure mincross then the number of iterations before entering the procedure is also bounded as above and after entering the procedure mincross the algorithm will stay in the procedure no more than a logarithmic number of iterations before halting.
REFERENCES
