SOME ASPECTS OF H.W. LENSTRA’s INTEGER PROGRAMMING ALGORITHM

by

Azaria Paz

Technical Report #359
March 1985
SOME ASPECTS OF H.W. LENSTRA'S INTEGER PROGRAMMING ALGORITHM

by

Azarja Paz

Computer Science Department
Technion - Israel Institute of Technology
Haifa 32000, Israel

ABSTRACT

An algorithm for integer programming, similar to H.W. Lenstra's algorithm but 'dual' to it, in a way to be described in the text is introduced. The algorithm has basically the same complexity as Lénstra's but is easier to implement. The second part of the paper deals with linear Diophantine equations as a particular case.
1. Introduction and Preliminaries

A couple of years ago, W.H. Lenstra Jr. [6, 1983] introduced a new and interesting algorithm for solving the following problem. Given $m \times n$ and $m \times 1$ matrices $A$ and $b$ respectively of integers, find whether there exists an $n \times 1$ vector of integers $x$ satisfying $Ax \leq b$ (entrywise). Lenstra showed that his algorithm is polynomial when the number of variables, $n$, is fixed. Since then several related works have been published [2, 7] and the algorithm has been used for investigating other problems in diophantine approximations and cryptography [2, 4, 8].

This work is based on work of W.H. Lenstra [6]. The reader is assumed to be familiar with that work but we will try to have this paper stand on its own. We shall use the definitions and notations below.

A convex set is a set of the form

$$K = \{x \in R^n, (1,x) B \geq 0\}$$

where $B$ is an $(n+1) \times m$ matrix of integers and $(1,x) = (1,x_n,x_{n-1},...,x_1)$. The number of variables involved ($n$) is the dimension of the set. $K$ can also be given as the convex hull of its vertices and those vertices can be found by solving at most $\lceil \frac{m}{n} \rceil \leq m^n$ systems of $n$ equations derived from $B$. Let $v_1, v_2, ..., v_l$ be the vertices of $K$, $l > 1$. Let $d$ be the dimension of the linear subspace generated by the vectors $v_j - v_0$, $1 \leq j \leq l$. Then $d$ will be called the rank of $K$. If $d = n$ then $K$ is full rank.

For a given convex set as defined above, Lenstra's algorithm decides that it contains no point with integral coordinates or produces such a point if it exists. We intend to introduce below an algorithm which is similar to Lenstra's and use a modification of Lenstra's algorithm in order to prove ours. We begin by describing a few procedures to be used in the final algorithm.
2. PROCEDURES AND FACTS

2.1 The Density of a General Lattice

Let \( b_1, \ldots, b_n \) be \( n \)-dimensional linearly independent point vectors with rational coordinates constituting the basis of an \( n \)-dimensional lattice to be denoted by \( L(b_1, \ldots, b_n) \) (i.e. the lattice points are all the combinations with integral coefficients of the basis vectors). The theorem below was proved by Lenstra in [6] and the reader is referred to that paper for its proof.

**Density Theorem:** For any point \( z \in \mathbb{R}^n \) one can find (constructively) a point \( y \in L(b_1, \ldots, b_n) \) such that

\[
|z - y| \leq \frac{1}{2} \sqrt{|b_1|^2 + \cdots + |b_n|^2}
\]

**Corollary 2.1:** Any sphere with radius \( r \) such that \( r \geq \frac{1}{2} \sqrt{\sum_{i=1}^{n} |b_i|^2} \) contains a lattice point. This lattice point and the center of the sphere are included in one basic parallelepiped of the lattice.

2.2 The \( L^3 \) algorithm

A basic subroutine in Lenstra's algorithm is the algorithm of A.K. Lenstra, H.W. Lenstra and L. Lovász [5] known in the literature as the \( L^3 \) algorithm. It does the following:

Given a general basis \( b_1, \ldots, b_n \) for an \( n \)-dimensional lattice, where the \( b_i \)'s have rational coefficients. It constructs in polynomial time (in \( n \) and \( |b_n| \) assuming that \( |b_1| \leq |b_2| \leq \cdots \leq |b_n| \)) another equivalent basis (i.e. spanning the same lattice) \( b'_1, b'_2, \ldots, b'_n \) which has several nice properties. The property we shall need here is the following

\[
|b_i| \leq 2^{n-1} h_n, \quad 1 \leq i \leq n
\]
where $h_n$ is the projection of $b_n$ perpendicular to the space spanned by $b_1, \ldots, b_{n-1}$. The above property implies that

$$
\sum_{i=1}^{n-1} |b_i|^2 \leq 2 \frac{n-1}{2} \sqrt{n} |h_n|
$$

and this, together with the Corollary 2.1 in the previous section induces the following.

**Corollary 2.2:** Any sphere with radius $r$ such that

$$
r \geq \frac{1}{2} \cdot 2^{\frac{n-1}{2}} \sqrt{n} |h_n|
$$

contains a lattice point. The lattice point and the center of the sphere are included in one basic parallelepiped of the lattice.


### 2.3 Spheres and Straight Simplices in n-dimensional Geometry

Consider the $n$-dimensional 'straight' simplex whose vertices are the origin $(0, \ldots, 0)$ and the unit vector $(1,0,\ldots,0)\ldots(0,0,\ldots,1)$. We want to find the radius $r$ of the hypersphere inscribed in the above simplex. To evaluate $r$ consider the two-dimensional section of the simplex as depicted below.

![Figure 1](image)
The center of the hypersphere is equally distanced from the hyperplanes \( z_i = 0 \). Its location is therefore at the point \( (r, r, ..., r) \) on the line joining the origin with the point \( \left( \frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n} \right) \) on the simplex, where the hypersphere touches the 'slanted' face of the simplex (as follows from symmetry). The distance between the points \( (r, r, ..., r) \) and \( \left( \frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n} \right) \) is therefore \( r \) which implies that

\[
\sqrt{n \left( \frac{1}{n} - r \right)^2} = r \quad \text{or} \quad \sqrt{n \left( \frac{1}{n} - r \right)} = r
\]

Thus

\[
r \left( 1 + \sqrt{n} \right) = \frac{\sqrt{n}}{n} = \frac{1}{\sqrt{n}} \quad \Rightarrow \quad r = \frac{1}{\sqrt{n} \left( 1 + \sqrt{n} \right)}.
\]

3. THE INTEGER PROGRAMMING ALGORITHM

H.W. Lenstra's integer programming algorithm reduces an \( n \)-dimensional integer programming problem to several \( (n-1) \)-dimensional such problems and then proceeds recursively showing that the total number of one-dimensional problems to be solved is polynomial in the length of the input for a fixed number of variables.

Given a convex \( n \)-dimensional set in the form

\[
K = \{ x \in \mathbb{R}^n; (1, x) B \geq 0 \}
\]

as defined in the introduction, Lenstra's reduction process can be divided into 3 stages. The first two stages are polynomial time (in the number of variables and in the length of the input) preconditioning stages which we would like to take for granted. Still for the sake of completeness we present a brief description of those stages.
3.1 The First Stage

Assume that $K$ is given in $n$-dimensional Euclidean space and assume that $K$ is bounded. Otherwise, one can add inequalities to the inequalities defining $K$ and get a bounded set $K'$ such that $K' \cap \mathbb{Z}^n$ is not empty if and only if $K \cap \mathbb{Z}^n$ is not empty (see e.g. [9]).

Given that $K$ is bounded one finds $n+1$ vertices, $v_0,v_1,...,v_n$ of $K$ such that the vectors $v_1-v_0, v_2-v_0, ..., v_n-v_0$ are linearly independent, provided that $K$ is not empty and provided that it is full rank. If the rank of $K$ is $d < n$ then an equivalent full rank $K'$ is constructed while finding the $d+1$ vertices as above. We shall assume therefore that $K$ is of rank $n$.

3.2 Second Stage

Let $\varepsilon$ be a given rational number $0 < \varepsilon \leq \frac{1}{2}$. The set of vertices of $K$, $v_0,...,v_n$ found in the first stage is replaced by a new set of vertices $v'_0,v'_1,...,v'_n$, such that the vectors $v'_1-v'_0,v'_2-v'_0,...,v'_n-v'_0$ are linearly independent and have the following property:

Let $H_i$ be the hyperplane spanned by the vertices $v'_0,...,v'_i-1,v'_i+1,...,v'_n$. Let $0 < d(u_i,H_i)$ be the distance of $u_i$ from $H_i$, and let $u_i$ be the point located on the line passing through $u_i$ and perpendicular to $H_i$, on the same side of $H_i$ as $u_i$, and such that $d(u_i,H_i) = (1+\varepsilon)d(u_i,H_i)$. Let $H'_i$ be the hyperplane passing through $u_i$ and parallel to $H_i$. Finally, let $u'_i$ be the intersection point of the hyperplanes $H'_0,H'_1,...,H'_{i-1},H'_{i+1},...,H'_n$. Then $K$ is inside the convex hull of the vertices $(v'_0,v'_1,...,v'_n)$. The figure below shows the configuration for the 2-dimensional case. It is easy to show that if $\varepsilon$ is rational and the $v'_i$ are rational vectors then so are the vectors $u_i$ and $u'_i$.

To simplify the notation we shall relabel in the sequel $v'_n$ by $v'_1$ and $v'_i$ by $u_i$ for $0 \leq i \leq n$. 

Technion - Computer Science Department - Technical Report CS0359 - 1985
3.3 The Reduction Stage

Based on the previous two stages, we shall assume here that the given convex set $K$ is bounded, is of full rank and that we have already found vertices $v_0, v_1, \ldots, v_n$ and corresponding vertices $v_0', v_1', \ldots, v_n'$ satisfying the conditions set in the second stage, i.e. the vectors $v_1 - v_0, \ldots, v_n - v_0$ are linearly independent and

$$\text{Conv}(v_0, \ldots, v_n) \subset K \subset \text{Conv}(v_0', \ldots, v_n').$$

Sometimes the problem is already given in this form (i.e. when $K = \text{Conv}(v_0, \ldots, v_n)$). If such is the case, then the preconditioning stage can be avoided altogether.

Our reduction procedure will be given as a sequence of 4 simple transformations. The first two transformations are equivalent to Lenstra's, and include a slight but significant modification rendering the computations involved simple and efficient. The last two transformations will set the problem into a simple form easy for implementation (for moderately many variables).

3.3.1 First Transformation

Let $\hat{V}$ be the $(n+1) \times n$ matrix whose rows are the vectors $v_0, \ldots, v_n$ and let $\hat{\hat{V}}$ be the $n \times n$ matrix whose rows are the vectors $v_1 - v_0, \ldots, v_n - v_0$. Compute $\hat{\hat{V}}^{-1}$. By definition $\hat{V}$ is nonsingular with rational entries, therefore $\hat{\hat{V}}^{-1}$ has the same
The following notation will be used in the sequel. The convex set (simplex) spanned by the rows of an \( (n+1) \times n \) matrix \( V \) will be denoted by the same notation \( V \) (the ambiguity of the notation will be resolved by the context). If \( K \) is a convex set and \( T \) is a transformation then \( KT \) denotes the transformed convex set. In particular if \( V \) denotes a convex set as above then \( VT \) denotes the convex set spanned by the rows of the transformed matrix \( VT \).

The first transformation is achieved by moving every point \((x_1, \ldots, x_n)\) in space to the point \((y_1, \ldots, y_n) = (x_1, \ldots, x_n) \hat{V}^{-1}\). Then

\[
W \hat{V}^{-1} \subseteq \hat{K} \hat{V}^{-1} \subseteq \hat{V} V^{-1},
\]

where \( V \) is the matrix with rational entries whose rows are the vectors \( v_0', \ldots, v_n' \).

Consider the unit matrix \( I \) in the original space to be the matrix whose rows are a basis for the natural lattice. The original problem is now transformed into the problem: Find a lattice point of the lattice spanned by the rows of the matrix \( IV^{-1} = \hat{V}^{-1} \) inside the convex set \( K \hat{V}^{-1} \), or decide that no such point exists. Notice that, as follows from the linearity of the transformation, the two convex sets defined by \( W \hat{V}^{-1} \) and \( V' \hat{V}^{-1} \) remain parallel one to the other (i.e. corresponding hyperplanes bordering those sets are parallel one to another). It follows from the transformation defined above that the simplices defined by \( W \hat{V}^{-1} \) and \( V' \hat{V}^{-1} \) are 'straight', i.e. translates of a simplex with one vertex at the origin and the other vertices equally distant from the origin and along the coordinate axes. It is easy to see that the projection of those simplices on the \( i,j \)-coordinate hyperplane has the form depicted below (\( \hat{v}_i \) and \( \hat{v}_j \), etc. denote the projection of the corresponding vertices after transformation).

\[
\hat{v}_i - \hat{v}_0 = (0 \ldots 1 \ldots 0) \text{ which implies that } |\hat{v}_i' - \hat{v}_0'| = (0 \ldots 2+3\varepsilon \ldots 0) \text{ by the definitions.}
\]
The ratio between the linear dimensions of the two sets is therefore equal to $2 + 3\varepsilon$.

3.3.2 Second Transformation

Apply the $L^3$ algorithm to the rows of the matrix $\hat{V}^{-1}$ (constituting a basis of the transformed lattice), to get a new basis $B = L\hat{V}^{-1}$ for the same lattice, where $L$ denotes the unimodular matrix of the transformation defined by the $L^3$ algorithm (Section 2.2). Denote by $b_1...b_n$ the rows of $B$. The simplex $\hat{V}^{-1}$ is 'straight' so that the derivation in Section 2.3 applies.

Let $r$ be defined as in Section 2.3. We distinguish two cases. Either $r = \frac{1}{\sqrt{n}(\sqrt{n} + 1)} \geq \frac{1}{2} \sqrt{\sum_{i=1}^{n} |b_i|^2}$ (Section 2.3 and Corollary 2.1) then the problem has a solution, which can be found as described in [6]. Or $r < \frac{1}{2} \sqrt{\sum_{i=2}^{n} |b_i|^2}$, in which case the number $t$ of hyperplanes perpendicular to $b_n$ which cut the outer simplex is 'small' ($O(n^{3/2})$ as shown in [6]).

The $n$-dimensional problem can now be reduced to at most $O(n^{3/2})$ problems of dimension $n-1$. 
So far, as mentioned at the beginning of Section 3, these two transformations together with the preconditioning are equivalent to Lenstra's algorithm up to the following change:

The first transformation (3.3.1) is very simple, easy to implement and involves rational numbers only (in contradiction to the corresponding transformation used in Lenstra's algorithm). On the other hand, the bound on the number of subdivisions remains basically the same. This would be easy to show but we want to change the algorithm first and introduce some additional simplifications, which will also render the bound on the number of subdivisions slightly smaller. This is done next.

3.3.3 Third Transformation

After the second transformation, we are left with a lattice whose base are the rows of the matrix $B = L\hat{V}^{-1}$, where $L$ is unimodular and with the configuration

$$\hat{W}^{-1} \subseteq K\hat{V}^{-1} \subseteq \hat{V}\hat{V}^{-1}. \quad (5)$$

We return now to the original space by applying the transformation $\hat{V}$. This will change the lattice into the natural lattice but spanned by the rows of the matrix $L = B\hat{V} = L\hat{V}^{-1}\hat{V}$, whose entries are integers. The configuration (5) above is changed into the configuration

$$V \subseteq K \subseteq \mathcal{V} \quad (6)$$

as was originally set.

3.3.4 Fourth Transformation

We apply now the transformation $L^{-1}$ to the whole space. $L^{-1}$, being unimodular preserves the natural lattice, but the new basis is changed into the trivial basis ($LL^{-1} = I$). The configuration (6) is now changed to
Now, $L^{-1}$ represents a linear transformation. This implies that the convex set defined by $VL^{-1}$ is parallel to the convex set defined by $V_{L}^{-1}$ (corresponding hyperplanes bordering those sets are parallel to one another) with the ratio between the linear dimensions of those sets preserved and equal to $2 + 3\varepsilon$ (see figure 3).

Consider the two matrices $L^{\hat{V}}^{-1}$ and $V_{L}^{-1}$ and denote the rows of the first matrix and the columns of the second matrix by $b_1, b_n$ and \( \hat{b}_1, \ldots, \hat{b}_n \) respectively. Let $h_n$ be the projection of $b_n$ perpendicular to the space spanned by the other $b_i$'s. $\hat{b}_n$ is perpendicular to the vectors $b_1, \ldots, b_{n-1}$ and is therefore parallel to $h_n$. It follows that $(\hat{b}_n, b_n) = (\hat{b}_n, h_n) = |\hat{b}_n| |h_n| = 1$.

Thus $|\hat{b}_n| = \frac{1}{|h_n|}$.

By Corollary 2.2, if $r = \frac{1}{\sqrt{n}(\sqrt{n}+1)} \geq \frac{1}{2} 2^{\frac{n-1}{2}} \sqrt{n} |h_n|$ then a solution can be found as shown in [6] ($r$ is defined as in Section 2.3). Otherwise,

$|h_n| > \frac{\sqrt{n}}{2^{\frac{n-1}{2}} n(\sqrt{n}+1) \cdot 2^{\frac{n-3}{2}}}$ implying that $|\hat{b}_n| = \frac{1}{|h_n|} < n(\sqrt{n}+1) \cdot 2^{\frac{n-3}{2}}$.

Thus $V_{L}^{-1}$ has a column vector shorter than $2^{\frac{n-3}{2}} n(\sqrt{n}+1)$. This implies that $V_{L}^{-1}$ has a column vector $\xi = (\xi_i)$ such that $|\xi_i - \xi_0| \leq 2^{\frac{n-3}{2}} n(\sqrt{n}+1)$ for all $i$, and therefore $|\xi_i - \xi_j| \leq 2^{\frac{n-1}{2}} n(\sqrt{n}+1)$ for all $i$ and $j$. But the linear dimension of the convex set defined by $VL^{-1}$ is larger than the linear dimension of the set $VL^{-1}$ by a factor of $(2 + 3\varepsilon)$ which shows that the matrix $VL^{-1}$ contains a column vector $\xi' = (\xi'_i)$ such that

$$|\xi'_i - \xi'_j| < (2 + 3\varepsilon) \cdot 2^{\frac{n-1}{2}} n(\sqrt{n}+1) \cdot 2^{\frac{n-1}{2}} n(\sqrt{n}+1) \cdot 2^{\frac{n-3}{2}}.$$
Slicing the set $V'L^{-1}$ into slices perpendicular to the coordinate corresponding to $\xi$ will therefore reduce the problem into $t$ (as defined in (8)) subproblems of dimension $n-1$, already given in $(n-1)$-dimensional space.

Remark: The above argument shows the possibility of a reduction algorithm which is, in a sense, dual to H.W. Lenstra's as described below.

Given $V$ and $V'$ as defined in Section 3.3: Consider the columns of $V$ as a basis of a (dual) lattice. Find a unimodular transformation $T$ transforming the basis $V$ into a new basis $VT = V'_1$ such that $V'_1$ contains a column $\xi$ whose maximal coordinate is minimal in absolute value. Then slice the convex set defined by $V'T$ by slices perpendicular to the coordinate defined by $\xi$. The first part of this problem is similar to the problem of finding the shortest vector in a lattice (see [2],[7]) and can be handled via the $L^3$ algorithm. The number of resulting $(n-1)$-dimensional slices from the dual algorithm is smaller or equal than the number of slices produced by the direct algorithm.

3.3.5 Summing up

We can now summarize our version of H.W. Lenstra's algorithm as follows.

0. Input: $K$ is a bounded nonempty full rank convex set as defined in (1).

1. Preconditioning: Find two simplices with vertices $v_0,\ldots,v_n$ and $v'_0,\ldots,v'_n$ as described in Sections 3.1, and 3.2. Let $V$ and $V'$ be matrices whose rows are the above vertices correspondingly, and denote by $\hat{V}$ and $\hat{V}'$ the matrices whose rows are $v_i-v_0$ and $v'_i-v'_0$ correspondingly.

2. Find $\hat{V}^{-1}$ and apply $L^3$ to its rows. Denote by $L$ the unimodular matrix representing the $L^3$ transformation. Compute $L^{-1}$ (also unimodular).

3. If $b_1,b_2,\ldots,b_n$, the row vectors in $L\hat{V}^{-1}$ have the property that $\sqrt{\sum |b_i|^2} \leq \frac{2}{\sqrt{n}(\sqrt{n}+1)}$; then the solution can be found as shown in [5], but see also the remark below.
4. If \( \sqrt{\sum |b_i|^2} > \frac{2}{\sqrt{n(1+\sqrt{n})}} \), then transform the original space using the transformation \( L^{-1} \) which leaves the natural lattice invariant. The simplex \( V L^{-1} \) can now be subdivided into at most \((2+3\varepsilon)2^{\frac{n-1}{2}} n(\sqrt{n}+1)\) slices perpendicular to some coordinate axis (e.g. the slices are already given as \((n-1)\)-dimensional sets), where \( 0 \leq \varepsilon \leq \frac{1}{2} \) is defined as in Section 3.2.

5. Apply the algorithm recursively to the resulting slices.

Remark: Consider again step 3 above. If \( \sqrt{\sum |b_i|^2} \leq \frac{2}{\sqrt{n(1+\sqrt{n})}} \), then we know that a solution exists. Moreover, that solution is a vertex of a parallelepiped which is a translate of the basic parallelepiped spanned by \( b_1, \ldots, b_n \) and contains the center of the hypersphere inscribed in the simplex \( V \). The center of this hypersphere is easily seen to be equal to \( v_0 V^{-1} + (r, \ldots, r) \) (see Section 2.3). The center and the above mentioned vertex are both inside \( V V^{-1} \). After applying the transformation \( V L^{-1} \) the center of the hypersphere is transformed into the point \( v_0 V^{-1} + (r, \ldots, r) V L^{-1} \) and the parallelepiped is transformed into a unit hypercube whose vertices have integral coordinates and such that one of its vertices is inside the simplex. Steps 3 and 4 can therefore be replaced by the following:

3'. Compute the point \( v_0 V^{-1} + (r, \ldots, r) V V^{-1} = (x_1, \ldots, x_n) \). Let \( p_i = \lfloor x_i \rfloor \) and \( q_i = \lceil x_i \rceil \).

If one of the \( 2^n \) points \((s_1, \ldots, s_n)\) where \( s_i \in \{p_i, q_i\} \) is inside the \( V V^{-1} \), then we are done. If \((s_1, \ldots, s_n)\) is such a point then \((s_1, \ldots, s_n) L \) solves the original problem. Otherwise

4'. If step 3' fails then we know that \( \sqrt{\sum |b_i|^2} > \frac{2}{\sqrt{n(1+\sqrt{n})}} \) and therefore the simplex \( V L^{-1} \) can be subdivided into at most \((2+3\varepsilon)2^{\frac{n-1}{2}} n(\sqrt{n}+1)\) slices perpendicular to some coordinate axis already given as \((n-1)\)-dimensional sets.

The following features of our algorithm are worth noticing:

1. The complexity of the algorithm is basically the same as H.W. Lestra's original
algorithm and needs no further discussion.

2. All the calculations involved are rational or integral and no approximations are needed.

3. The subdividing is done in the natural lattice spanned by its trivial basis.

4. The subproblems resulting after the subdivision are already given as \((n-1)\)-dimensional problems (no extra transformations needed).

5. Any reduction in the coefficient \(2^{\frac{n-1}{2}}\) in Section 2.2 achieved by improving the \(L^3\) algorithm will improve the bound in step 4' of this algorithm.

6. The resulting \((n-1)\)-dimensional slices, define similar and parallel \((n-1)\)-dimensional sets. All those sets are defined by systems of inequalities which differ in their constant terms only. This parallelism inherent in the problem is thus set in a transparent form easy to take advantage of, provided that proper computational tools are available.

7. The number of inequalities defining all the resulting subproblems during the computation is bounded by the number of inequalities defining the original problem, thus bounding the number of vertices of the convex sets resulting from the subdivisions of the original problem.

4. A LINEAR DIOPHANTINE EQUATIONS

A particular and interesting case of the general integer programming problem is the problem of solving an equation of the form

\[ a_1x_1 + a_2x_2 + \ldots + a_nx_n = M \]  \hspace{1cm} (9)

over the nonnegative integers where the \(a_i\)'s and \(M\) are positive integers and \(gcd(a_1, \ldots, a_n) = 1\).

Before we proceed with the investigation of this problem we would like first to introduce two auxiliary algorithms we shall need here but may have additional
applications elsewhere.

4.1 Algorithms $U$ (unimodular) and $S$ (slice).

Let $\eta = (a_1, \ldots, a_n)$ be a vector of positive integers such that $\gcd(a_1, \ldots, a_n) = 1$. The algorithm $U$ below produces, in linear time, two matrices $A_\eta$ and $A_\eta^{-1}$, which are unimodular (i.e., have integral entries and their determinant equals to 1 in absolute value) and $A_\eta^{-1}$ has a column equal to $\eta^T$.

4.1.1 Algorithm $U$ (unimodular)

Input a vector $\eta = (a_1, \ldots, a_n)$ of positive integers

1. Set $U^{(0)} := V^{(0)} := I; \quad \eta^{(0)} := (a_1^{(0)}, \ldots, a_n^{(0)}) = \eta; \quad j := 0.$

Denote the $i$-th row and the $k$-th column of a matrix $U$ by $U[i,-]$ and $U[-,k]$ respectively.

2. While $\eta^{(j)}$ is not a unit vector do begin

2a. Choose, according to some preassigned version of the $n$-dimensional Euclidean algorithm (see e.g. [1]) two nonzero entries in $\eta^{(j)}$: $a^{(j)}_s \geq a^{(j)}_t$.

2b. Compute $\left| \frac{a^{(j)}_s}{a^{(j)}_t} \right| = k$.

2c. Set $a^{(j)}_s := a^{(j)}_s - k a^{(j)}_t; \quad a^{(j)}_t := a^{(j)}_t$ if $r \neq s$.

2d. Set $U^{(j)}[s,-] := U^{(j)}[s,-] - k U^{(j)}[t,-]$;

$U^{(j)}[r,-] := U^{(j)}[r,-]$ if $r \neq s$.

2e. Set $V^{(j)}[-,t] := V^{(j)}[-,t] + k V^{(j)}[-,s]$;

$V^{(j)}[-,r] := V^{(j)}[-,r]$ if $r \neq t$.

2f. $j := j + 1$

end while.

3. Output $A_\eta = U^{(j)}; A_\eta^{-1} := V^{(j)}.$

End of algorithm.
We shall assume w.l.o.g. that at termination \( \eta^{(j)} = (1,0,\ldots,0) \).

The following properties are easily proven and are left to the reader.

1. \( U^{(i)}, \gamma^{(i)} = I \) for \( 0 \leq i \leq j \) (the \( j \)-th iteration is assumed to be the last one), and both \( U^{(i)} \) and \( \gamma^{(i)} \) are unimodular.

2. \( U^{(i)} \eta^T = \eta^{(i)} \gamma^T \); \( \gamma^{(i)} \eta^{(i)} = \eta^T \) for \( 0 \leq i \leq j \).

3. \( \gamma^{(j)} = \eta^T \).

The first two properties follow by induction. The third property which is implied by the second is shown as follows:

\[
\eta^T = \gamma^{(j)} [U^{(j)} \eta^T] = \gamma^{(j)} \gamma^{(j)} \gamma^T = \gamma^{(j)} (1,0,\ldots,0)^T = \gamma^{(j)} [-,1].
\]

The complexity of this algorithm is the same as the complexity of the \( n \)-dimensional Euclidean algorithm, i.e. it is linear in the length of the input and in the dimension of \( \eta \). Notice that the entries in the matrix \( A_\eta^{-1} \) are nonnegative as follows from its construction.

4.1.2 Example 1

Given the vector \( \eta = (15,7,2) \)

1. Set \( \eta^0 = \eta \), \( U^{(0)} = \gamma^{(0)} = I \).

2a. Choose \( a_s^{(i)} \) to be the biggest and \( a_t^{(i)} \) to be the second biggest or equal entry in \( \eta^{(i)} \). If \( a_s^{(i)} \) is not unique, then choose the maximal \( s \) such that \( a_s^{(i)} \) is the biggest and similarly for \( a_t^{(i)} \). Then the first iteration results in:

\( a_s^{(0)} = 15 \), \( a_t^{(0)} = 7 \); \( k = 2 \); \( \eta^{(1)} = (1,7,2) \)

\[ U^{(1)} = \begin{bmatrix}
1 & -2 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}, \quad \gamma^{(1)} = \begin{bmatrix}
1 & 2 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix} \]

Second iteration:

\( a_s^{(1)} = 7 \); \( a_t^{(1)} = 2 \); \( k = 3 \); \( \eta^{(2)} = (1,1,2) \)
$$U^{(2)} = \begin{bmatrix} 1 & -2 & 0 \\ 0 & 1 & -3 \\ 0 & 0 & 1 \end{bmatrix} \quad V^{(2)} = \begin{bmatrix} 1 & 2 & 6 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix}$$

Third iteration:

$$a_1^{(3)} = a_2^{(3)} = 2; \quad a_2^{(3)} = 1; \quad k = 2; \quad \eta^{(3)} = (1,1,0)$$

$$U^{(3)} = \begin{bmatrix} 1 & -2 & 0 \\ 0 & 1 & -3 \\ 0 & -2 & 7 \end{bmatrix} \quad V^{(3)} = \begin{bmatrix} 1 & 14 & 6 \\ 0 & 7 & 3 \\ 0 & 2 & 1 \end{bmatrix}$$

Final iteration:

$$a_1^{(3)} = a_1^{(4)} = 1; \quad a_2^{(3)} = a_2^{(4)} = 1; \quad k = 1; \quad \eta^{(4)} = (1,0,0)$$

$$U^{(4)} = \begin{bmatrix} 1 & -2 & 0 \\ -1 & 3 & -3 \\ 0 & -2 & 7 \end{bmatrix} \quad V^{(4)} = \begin{bmatrix} 15 & 14 & 6 \\ 7 & 7 & 3 \\ 2 & 2 & 1 \end{bmatrix}$$

4.1.3 Algorithm $S$ (slice)

Given a convex $n$-dimensional set $K$ and an $n$-dimensional hyperplane $P$ we want to find an $(n-1)$-dimensional convex set $K_1$ and an invertible mapping $\varphi$ such that: $y \in K_1$ and $y$ has integral coordinates if and only if $x = \varphi(y) \in K \cap P$ and $\varphi(y)$ has integral coordinates. Thus $K_1$ will represent a 'slice' of $K$ by $P$. The algorithm is described below. We assume first that the coefficients of $P$ are positive, a restriction to be removed in the sequel.

1. Input $K = \{x \in \mathbb{R}^n : (1,x)B \geq 0\}$, where $B$ is an $(n-1)\times m$ matrix of integers.

$$P = \{x \in \mathbb{R}^n : x\eta^T = M\}$$

where $\eta = (a_1, \ldots, a_n)$ is a vector of positive integers and $M$ is an integer.

2. Apply algorithm $U$ to the vector $\eta$ resulting in $A_\eta$ and $A_\eta^{-1}$, $n \times n$ matrices with $A_\eta^{-1}[-1,1] = \eta^T$.

3. Construct the matrix $B'$ as below
where $B$ is defined in the definition of $K$. $B'$ is $(n+1)\times m$ as is $B$.

4. Construct the matrix $B_1$ as below

$$
B_1 = \begin{bmatrix}
1 & M & 0 & \ldots & 0 \\
0 & 0 & & & \vdots \\
0 & 0 & 0 & & I_{n-1} \\
\end{bmatrix}
$$

where $M$ is defined as in $P$, and $I_{n-1}$ is the $n-1$-dimensional unit matrix. $B_1$ is $n \times m$.

5. Set

$$
K_1 = \{ y \in \mathbb{R}^{n-1}: (1, y)B_1 \geq 0 \}.
$$

6. The transformation $\phi$ mapping $K \cap P$ onto $K_1$ is defined as follows:

for $x \in K \cap P$, set $xA_\eta^{-1} = (M, \phi(x))$ (recall that $A_\eta^{-1}[-,1] = \eta^\top$ so that $x \in P$ implies that $xA_\eta^{-1}[-,1] = x\eta^\top = M$). Set $\phi(x) = y$. $\phi^{-1}(y)$ can be computed by $\phi^{-1}(y) = (M, y)A_\eta$, and $\phi(x)$ can be computed by $\phi(x) = x(A_\eta^{-1}[-,2], \ldots, A_\eta^{-1}[-,n])$.

**Proof of Algorithm:**

Assume first that $x \in K \cap P$. Then $x \in P$ implies that $xA_\eta^{-1} = (M, \phi(x))$ which implies that $x = (M, \phi(x))A_\eta$. 


Now $x \in K$ implies, using the equality above, that

$$0 \leq (1,x)B = (1,(M,\varphi(x))A_\eta)B =
\begin{bmatrix}
1 & 0 & \cdots & 0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\begin{bmatrix}
1, M, \varphi(x) \\
\vdots \\
A_\eta \\
0
\end{bmatrix}
B = (1,M,\varphi(x))B' = (1,\varphi(x))B_1.
$$

Conversely, if $y \in K_1$ then clearly $x = (M,y)A_\eta$ has the property that $\varphi(x) = y$ and, as $xA_\eta^{-1} = (M,y)$, we have that $x\eta^T = xA_\eta^{-1}[-1,\ldots,-1] = M$ implying that $x \in P$. In addition, we have also that $0 \leq (1,y)B_1 = (1,M,y)B' = (1,(M,y)A_\eta)B = (1,x)B$ implying that $x \in K$. Thus $x = \varphi^{-1}(y) = (M,y)A_\eta \in P \cap K$ as required.

It is clear from the construction that $x$ has integral coordinates if and only if $y$ has, but the transformation defined maps the whole set $K \cap P$ onto the set $K_1$.

**Remark.** If the hyperplane $P$ is defined by a vector $\gamma$ with some negative or zero coordinates, the algorithm can still be applied as follows:

If the $i$-th entry of $\gamma$ is zero then set the $i$-th row of $A_\eta$ and the $i$-th column of $A_\eta^{-1}$ to be the $i$-th coordinate unit row and column vectors respectively (i.e. $(0 \ldots 010 \ldots 0)$ and its transpose) and apply algorithm $U$ to the other entries of $\gamma$. If some entries of $\gamma$ are negative then apply algorithm $U$ to $\gamma' = (|a_1|,|a_2|\ldots|a_n|)$, and then change the resulting matrices $A_\eta$ and $A_\eta^{-1}$, multiplying the columns (rows), of $A_\eta(A_\eta^{-1})$, corresponding to negative values of $\gamma$, by $-1$.

The proof that the algorithm is correct for this case too is left to the reader.
4.1.4 Example 2

Let $P$ be the plane defined by

$$50549x_1 + 140701x_2 + 49217y_3 = 42,275,139.$$  

Let $K$ be defined by $K = \{(x_1,x_2,x_3): x_i \geq 0\}$, then

$$y = (50549, 140701, 49217) = (a_1,a_2,a_3); \quad M = 42,275,139$$

and

$$B = \begin{bmatrix} 0 & 0 & 0 \\ I_3 \end{bmatrix}$$

(it is easy to see that $\{x: x_i \geq 0\} = \{x: (1,x)B \geq 0\}$).

Applying algorithm $U$ to the vector $y$ we get the matrices

$$A_\eta = \begin{bmatrix} 23 & -31 & 65 \\ 41 & 107 & -384 \\ -391 & 180 & -113 \end{bmatrix}; \quad A_\eta^{-1} = \begin{bmatrix} 50549 & 8197 & 3833 \\ 140701 & 22816 & 10899 \\ 49217 & 7931 & 3732 \end{bmatrix}$$

Due to the special form of $B$ in this example, we get that

$$B' = \begin{bmatrix} 0 & 0 & 0 \\ \hline A_\eta \end{bmatrix} \quad \text{and} \quad B_1 = \begin{bmatrix} M & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}A_\eta$$

i.e. $B_1$ can be derived from $A_\eta$ by multiplying its first row by $M$. The equivalent two dimensional set is therefore

$$K_1 = \{y: (M,y)A_\eta \geq 0\}.$$ 

If $y = (y_1,y_2) \notin K$, then $x = (M,y)A_\eta^{-1}$ is in $K \cap P$, i.e. $x$ is a solution of the equation defining $P$ over the nonnegative integers.

The following interesting feature of the above transformation is worth noticing.

The vertices of the 3-dimensional triangle defined by $P \cap K$ are $(M/a_1,0,0), (0,M/a_2,0), (0,0,M/a_3)$ when $\eta = (a_1,a_2,a_3)$ is as given at input. The area of the projection of this triangle on the $x_1,x_2$ plane is $\frac{M}{a_1} \cdot \frac{M}{a_2} \cdot \frac{1}{2}$ whose integral
part is equal to 125,615. Thus the number of feasible $x_1, x_2$ integer values corresponding to possible nonnegative integral solutions of the equation $x\eta^T = M$ is quite big. But the vertices of the triangle defined by $K_1$ are easily found by applying the transformation $\varphi$ to the vertices of $P \cap K$, that is multiplying the point vectors $(M/a_1, 0, 0), (0, M/a_2, 0), (0, 0, M/a_3)$ by $A_\eta$. The resulting $(y_1, y_2)$ vertices are found to be

\begin{align*}
  y_1 & = \frac{M}{2} + 197/50549, \\
  y_2 & = \frac{N}{2} + 33/50549, \\
  y_2 & = \frac{M}{2} + 22816/140701, \\
  y_3 & = \frac{N}{2} + 22816/140701, \\
  y_4 & = \frac{N}{2} + 7921/49217, \\
  y_5 & = \frac{N}{2} + 3732/49217.
\end{align*}

Now,

$$\left| \max y_1 - \min y_1 \right| = 6855314 - 6855312 = 2$$

and

$$\left| \max y_2 - \min y_2 \right| = 3205616 - 3205615 = 1.$$

Thus, the number of integral points possibly inside $K_1$ is equal to 6 (!) and out of those 6 points 2 are actually inside $K_1$ inducing the 2 solutions below to the original equation (via $(x_1, x_2, x_3) = (M, y_1, y_2)A_\eta^{-1}$)

\begin{align*}
  x_1 & = 174, \\
  x_2 & = 62, \\
  x_3 & = 503.
\end{align*}

Several other examples we have tried resulted in similar dramatic 'compactizations'. This phenomenon will be discussed in the sequel.

### 4.2 Reduction to Integer Programming

As in Example 2 of the previous section any linear diophantine equation can be reduced to an $n-1$-dimensional integer programming problem such that the resulting convex set is a simplex. This is done as follows.

Given the linear equation
Construct the matrices $A_\eta$ and $A_\eta^{-1}$ using algorithm $S$. The equivalent convex set is then

$$K = \{ y : (M,y)A_\eta \succeq 0 \}$$

(10)

If $x$ solves (9) then $y$, defined by $x A_\eta^{-1} = (M,y)$ is in $K$. Conversely, if $y$ is in $K$ then $x = (M,y)$ solves (9). Due to the fact that $K$ in (10) is defined by $n$ inequalities in $(n-1)$-dimensions, we have that $K$ has $\hat{n} = \left\lfloor \frac{n}{n-1} \right\rfloor$ vertices and is therefore a simplex. We can now slice $K$ into $(n-2)$-dimensional slices, etc., with no preconditioning involved in the first reduction of dimension, as shown in Section 3.

There are some additional interesting features of the above algorithm which are listed below.

Remark: The reduction described in this section can be extended to nonlinear convex sets. Assume e.g., that a nonlinear ellipsoidal set is defined as below

$$K' = \{ x : (1,x)A(1,x)^T \succeq 0 \}$$

where $A$ is a symmetric positive-matrix. Then the intersection of $K'$ with the equation (9) can be expressed as an equivalent $(n-1)$-dimensional ellipsoidal set $K_1$ defined below

$$K_1 = \{ y : (1,(M,y)A_\eta)A(1,(M,y)A_\eta)^T \succeq 0 \}$$

$$= \{ y : (1,y)A_1(1,y)^T \succeq 0 \}$$

with $A_1$ a symmetric positive matrix.

The equivalence between $K' \cap (9)$ and $K_1$ is given by

if $x \in K' \cap (9)$ then $x A_\eta = (M,y)$ and $y \in K_1$.

if $y \in K_1$ then $x = (M,y)A^{-1} \in K' \cap (9)$.

$x$ has integral entries iff $y$ has. The details are left to the reader.
4.2.1 The Vertices

As mentioned in the previous section the equation (9) reduces to an integer programming problem which is represented by an \((n-1)\)-dimensional simplex. The vertices of this simplex can be found as follows:

Let \( A_n^{-1} = [a_{ij}] \) with \( a_{ii} = a_i = A_n^{-1}[i, 1] \) for all \( i \). The vertices of the original set (9) are \( x(i) = (0, ..., M/a_i, ..., 0), 1 \leq i \leq n \). Those vertices are transformed into the vertices of \( K \) via the transformation

\[
y(i) = [0, ..., M/a_i, ..., 0] [A_n^{-1} [-2], ..., A_n^{-1} [-n]] = M/a_i [a_{i2}, ..., a_{in}]; 1 \leq i \leq n \quad (10)
\]

(recall that \( a_1 = a_i \)).

The \( i \)-th vertex of \( K \) is therefore proportional to the last \( n-1 \) entries of the \( i \)-th row of \( A_n^{-1} \) with coefficient of proportionally equal to \( M/a_i \).

Having found the vertices of \( K \) we can proceed as in Section 3, reshape \( K \) into \( K' = KL^{-1} \) and then slice \( K' \) (if a solution has not yet been found) into slices of the form \( K_j = ((x_i = c_j) \cap K') \) of dimension \( n-2 \), for some index \( i \) and constants \( c_j \). The vertices of \( K' \) are divided by the hyperplane \( x_i = c_j \) into 3 sets: vertices on, 'above', and 'below' the hyperplane. Any vertex of \( K_j \) is therefore either a vertex of \( K' \) on \( x_i = c_j \) or the intersection of this hyperplane with a line joining a vertex above it with a vertex below it. This suggests an easy method for finding vertices of \( K_j \).

Notice also that, as remarked before, the number of vertices of any subproblem, generated by the algorithm, never exceeds \( \left\lfloor \frac{n}{2} \right\rfloor \), this following from the fact that the number of inequalities defining the various convex sets involved in our algorithm never exceeds \( n \). The above facts may reduce the amount of work involved in the preconditioning stages of the algorithm.
4.2.2 Diophantine Approximation

The reduction of the linear equation (9) to an integer programming problem as given in Section 4.2 is not uniquely defined, due to the fact that the matrices $A_\eta$ and $A_\eta^{-1}$ generated by the algorithm $U$ depend on the choice of a version of the $n$-dimensional Euclidean algorithm, and, as shown in the previous section, the vertices of the resulting $(n-1)$-dimensional convex set are proportional to the rows of the matrix $A_\eta^{-1}$. Specifically, the $i$-th vertex of the resulting convex set $K$ is equal to $M/a_{i1}[a_{i2},...,a_{in}]$ as in (11) in the previous section. The number of slices of $K$ perpendicular to some coordinate $j$ will be 'small' if $\frac{\text{Max}}{\text{Min}} \frac{a_{ij}}{a_{11}} - \frac{a_{ij}}{a_{11}}$ is 'small' and this will happen if the vectors $A_\eta^{-1}[-,1]$ and $A_\eta^{-1}[-,j]$ are 'close' in the sense of Diophantine approximation (see [3]).

E.g. assume that $|\frac{a_{1j}}{a_{11}} - \frac{a_{pj}}{a_{p1}}| < \varepsilon$ for some $\varepsilon$ and all $p$. Then $|\frac{a_{1j}}{a_{11}} - \frac{a_{pj}}{a_{p1}}| < \frac{\varepsilon}{a_{p1}}$.

\[ \left| \frac{a_{pj}}{a_{p1}} - \frac{a_{qj}}{a_{q1}} \right| \leq \left| \frac{a_{pj}}{a_{p1}} - \frac{a_{1j}}{a_{11}} \right| + \left| \frac{a_{qj}}{a_{q1}} - \frac{a_{1j}}{a_{11}} \right| \leq \varepsilon \left( \frac{1}{a_{p1}} + \frac{1}{a_{q1}} \right) \]

for all $p$ and $q$. If such is the case then the number of slices of $K$ perpendicular to the $j$-th coordinate is less than $\varepsilon \left( \frac{1}{a_{p1}} + \frac{1}{a_{q1}} \right) M$.

As mentioned before, the algorithm $U$ producing the matrix $A_\eta^{-1}$ allows for variations in the choice of a proper version of an $n$-dimensional Euclidean algorithm. One should therefore choose a version such that the resulting matrix $A_\eta^{-1}$ will have its last $n-1$ columns as 'close' as possible in the Diophantine approximation sense as explained above, to its first column vector, which is the vector of the coefficients of the given equation (9). A good heuristics to achieve this goal seems to be the following: Choose a version of the $n$-dimensional Euclidean algorithm which will render the 'entries of $A_\eta$ as small as possible. This heuristic is based on many examples we tried. In a forthcoming paper we shall present a
probabilistic $n$-dimensional Euclidean algorithm for achieving the above goal.

Finding a matrix $A_{\eta}^{-1}$ such that its columns are as above, is therefore related to the finding of a Minkovsky-reduced basis for the lattice spanned by the edges of the convex set $K$ defined by the equation (9) (see [7]). A good heuristics for finding such a matrix $A_{\eta}^{-1}$ can often serve as a substitute for the basis reductions involved in the algorithm.

As a final remark we notice that after a linear diophantine equation has been solved for a given $M$, we can save most of our computations and use them for solving the same equation with another $M$. Solving again the same equation with a different $M$ would be much easier the second time as all the new geometrical configurations corresponding to the new $M$ are parallel to the configurations found for the first $M$.

4.2.3 Example 3:

The example below illustrates the discussion in the previous sections. We want to solve the equation

$$271x_4 + 277x_3 + 281x_2 + 283x_1 = 15000 = M$$

over the nonnegative integers.

The area of the projection of the above hyperplane on the positive orthant of the $x_3,x_2,x_1$ space equals to $M^3/6 \cdot 277 \cdot 281 \cdot 283 > 25,535$. A brute force search will require therefore about 25,535 probes. Applying algorithm $U$ to the vector $\eta = (271, 277, 281, 283)$ with the same version of the 4-dimensional Euclidean algorithm used in Example 2 we get the following matrices

$$A_{\eta}^{-1} = \begin{bmatrix} 271 & 45 & 45 & 135 \\ 277 & 46 & 46 & 138 \\ 281 & 46 & 47 & 140 \\ 283 & 46 & 47 & 141 \end{bmatrix}, \quad A_{\eta} = \begin{bmatrix} 46 & -45 & 0 & 0 \\ -1 & 2 & 0 & -1 \\ 0 & -1 & 3 & -2 \\ 93 & 90 & -1 & 1 \end{bmatrix}$$
The problem is thus reduced to the convex set

\[ K = \{ y : (M,y)A_\eta \geq 0 \} \]

The vertices of \( K \) are found from \( A_\eta^{-1} \). Notice the fact that the columns of \( A_\eta^{-1} \) are 'closed' one to another in the sense discussed in the text as listed below.

<table>
<thead>
<tr>
<th>( y_5 )</th>
<th>( y_2 )</th>
<th>( y_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v_1 )</td>
<td>( 45\cdot M/275 )</td>
<td>( 45\cdot M/277 )</td>
</tr>
<tr>
<td>( v_2 )</td>
<td>( 46\cdot M/277 )</td>
<td>( 46\cdot M/277 )</td>
</tr>
<tr>
<td>( v_3 )</td>
<td>( 47\cdot M/281 )</td>
<td>( 47\cdot M/281 )</td>
</tr>
<tr>
<td>( v_4 )</td>
<td>( 48\cdot M/283 )</td>
<td>( 47\cdot M/283 )</td>
</tr>
</tbody>
</table>

where the actual values of \( y_1 \) have been calculated up to one position after the decimal point.

It is evident that \( y_1 \) can assume only one integral value, namely \( y_1 = 7473 \), i.e., the 3-dimensional simplex has only one 2-dimensional slice which is

\[ (15000,y_3,y_2,7473)A_\eta \geq 0. \quad (12) \]

The inequalities defining (12) are easily found.

\begin{align*}
    y_3 & \leq 2484 \\
    2y_3 - y_2 & \geq 2430 \\
    y_2 & \geq 2491 \\
    y_3 + 2y_2 & \leq 2473.
\end{align*}

The vertices of this set are given below (the actual computation is left to the reader).

<table>
<thead>
<tr>
<th>( y_3 )</th>
<th>( y_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( v_1 )</td>
<td>2484</td>
</tr>
<tr>
<td>( v_2 )</td>
<td>2484</td>
</tr>
<tr>
<td>( v_3 )</td>
<td>2466.6</td>
</tr>
<tr>
<td>( v_4 )</td>
<td>2460.5</td>
</tr>
</tbody>
</table>

Thus \( 2491 \leq y_2 \leq 2503; \ 2461 \leq y_3 \leq 2484. \)

The total number of integral feasible points is \( 24 \times 13 = 312 \) (24 is the range of \( y_3 \).
and 13 is the range of $y_2$). Probing those 312 points we find 171 solutions (more than one solution per two probes). Any $(y_3, y_2)$ solution of (12) induces a solution $x = (x_4x_3x_2x_1)$ of the original equation as defined below:

$$x = (15000, y_3, y_2, 7473)$$

e.g.

$y_3 = 2470, y_2 = 2498$ and $y_1 = 7473$

induces the solution of the original equation:

$$x_4 = 14, x_3 = 12, x_2 = 21, x_1 = 7$$

where

$$14 \cdot 271 + 12 \cdot 277 + 21 \cdot 281 + 7 \cdot 283 = 15000.$$
REFERENCES


