8

Basis Reduction

8.1 Introduction

Many combinatorial search problems can be reduced to solving a matrix equation of the form

$$AU = B \tag{8.1}$$

for a $(0,1)$-valued column vector U. For example let $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ be the set of edges of a graph $\mathcal{G}$ with vertex set $\mathcal{V} = \{1, 2, \ldots, n\}$. If $\mathcal{T} = \{T_1, T_2, \ldots, T_N\}$ is the set of all triangles in the graph $\mathcal{G}$, then we may define the m by N matrix A by

$$A[i,j] = \begin{cases} 1 & \text{if } e_i \text{ is an edge of } T_j; \\ 0 & \text{if not.} \end{cases}$$

A $(0,1)$-valued solution U to the matrix equation $AU = B$, where $B = [1, 1, \ldots, 1]^T$, is an edge decomposition of $\mathcal{G}$ into triangles. If a decomposition into other subgraphs is desired, then $\mathcal{T}$ can be chosen to contain these types of subgraphs. In Section 4.5, the problem of choosing members from a given collection $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ of subsets of a set $\mathcal{R} = \{1, 2, \ldots, n\}$ such that they partition $\mathcal{R}$ is discussed. If the matrix A is defined by

$$A[i,j] = \begin{cases} 1 & \text{if } i \in S_j; \\ 0 & \text{if not,} \end{cases}$$

and $B = [1, 1, \ldots, 1]^T$, then this problem asks for the $(0,1)$-valued solutions U to Equation 8.1. Recall that matrix A is called an incidence matrix. When the collection of graphs or sets in the decomposition are assumed to have certain symmetries, the resulting incidence matrices are discussed in Section 6.6.1. These incidence matrices need not be $(0,1)$-valued matrices; see Example 6.12.

When the size of the incidence matrix A is large, backtracking approaches often fail to find solutions in reasonable time. To get around this problem, the heuristic

search techniques of Chapter 5 can be employed. Another successful technique for solving Equation 8.1 has been the method of *basis reduction*. It begins by first transforming Equation 8.1 to the optimization problem of finding a shortest vector in a lattice. Consider the matrix equation

$$\begin{bmatrix} I & \vec{0} \\ A & -B \end{bmatrix} \begin{bmatrix} U \\ 1 \end{bmatrix} = \begin{bmatrix} U \\ \vec{0} \end{bmatrix}, \tag{8.2}$$

where $\vec{0}$ denotes the 0 vector and I the identity matrix. A vector $\vec{b} \in \mathbb{R}^q$ is represented in the computer as a q-dimensional array, whose entries are the components of $\vec{b}$. The following proposition is evident:

LEMMA 8.1 *The $(0,1)$-valued column vector U solves Equation 8.1 if and only if it solves Equation 8.2.*

If $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p \in \mathbb{R}^q$ are real-valued vectors, and $\alpha_1, \alpha_2, \ldots, \alpha_p \in \mathbb{R}$, then

$$\vec{b} = \alpha_1\vec{b}_1 + \alpha_2\vec{b}_2 + \cdots + \alpha_p\vec{b}_p$$

is a *linear combination* of $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p$ over the real numbers. The set of all linear combinations of $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p$ is the vector space

$$\mathsf{Span}_{\mathbf{R}}(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p) = \{\alpha_1\vec{b}_1 + \alpha_2\vec{b}_2 + \cdots + \alpha_p\vec{b}_p : \alpha_i \in \mathbb{R},\ 1 \le i \le p\}.$$

If the coefficients $\alpha_1, \alpha_2, \ldots, \alpha_p$ are restricted to be integers, then

$$\mathsf{Span}_{\mathbf{Z}}(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p) = \{\alpha_1\vec{b}_1 + \alpha_2\vec{b}_2 + \cdots + \alpha_p\vec{b}_p : \alpha_i \in \mathbb{Z},\ 1 \le i \le p\}$$

is said to be the *lattice* spanned by $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p$.

The set of vectors $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p$ is said to be *linearly independent* if the only linear combination over the real numbers that is the zero vector is the one in which $\alpha_i = 0$ for all i, $1 \le i \le p$.

Let B be a q by p matrix with linearly independent columns: $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p$. Then

$$\mathcal{L} = \mathsf{Span}_{\mathbf{Z}}(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_p) = \{B\vec{x} : \vec{x} \in \mathbb{Z}^q\}$$

is the lattice with basis (the columns of) B. Not every lattice has a basis of linearly independent vectors. One example is the lattice spanned by the vectors

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \text{and} \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

None of these vectors can be written as an integer linear combination of the other two. Thus each is required in the span of the lattice. On the other hand,

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \end{bmatrix} - 3\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

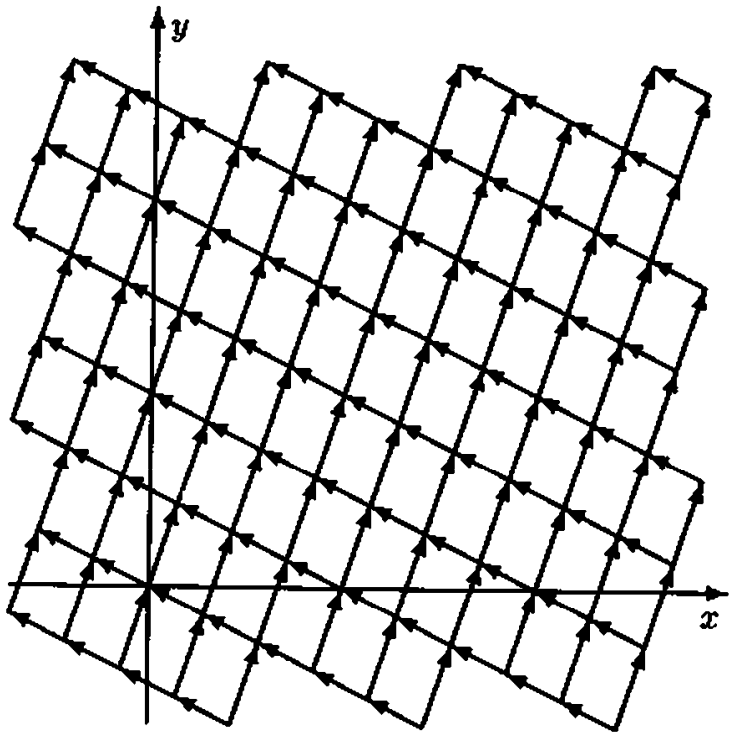
and so they are not linearly independent. We will only be interested in lattices that have a basis. If $p = q$ and the columns of B are linearly independent, then the lattice $\mathcal{L}$ with basis B is said to be a *full dimensional lattice*.

Example 8.1

Consider the matrix

$$B = \begin{bmatrix} 1 & -2 \\ 3 & 1 \end{bmatrix}.$$

The columns of B are the vectors $\vec{b}_1 = [1, 3]$, and $\vec{b}_2 = [-2, 1]$. It is easy to see that $\vec{b}_1$ and $\vec{b}_2$ are linearly independent. Thus the subspace $\mathrm{Span}_{\mathbb{R}}(\vec{b}_1, \vec{b}_2)$ is the entire xy-plane, but the lattice $\mathcal{L}$ with basis B is a discrete set of points. This is depicted by the following diagram.



□

Returning to Equation 8.2, we let $\mathcal{L}$ be the lattice whose basis is the $n + m$ by $n + 1$ matrix M defined by

$$M = \begin{bmatrix} I & \vec{0} \\ A & -B \end{bmatrix}.$$

Then Lemma 8.1 shows that if U is a solution to Equation 8.1, then

$$\widehat{U} = \begin{bmatrix} U \\ \vec{0} \end{bmatrix} \in \mathcal{L}.$$

Conversely, if

$$\vec{y} = [u_1, u_2, \ldots, u_n, \underbrace{0, 0, \ldots 0}_{m}]^T \in \mathcal{L},$$

then $\vec{y} = M\vec{x}$ for some integer valued column vector $\vec{x}$. Consequently, from Equation 8.2, it follows that

$$\vec{x} = [u_1, u_2, \ldots, u_n, d]$$

for some integer d. We record this result as Lemma 8.2.

LEMMA 8.2 *Let $\mathcal{L}$ be the lattice with basis*

$$M = \begin{bmatrix} I & \vec{0} \\ A & -B \end{bmatrix}.$$

Then there is a $(0,1)$-valued column vector

$$\vec{y} = [u_1, u_2, \ldots, u_n, \underbrace{0, 0, \ldots 0}_{m}]^T \in \mathcal{L} \tag{8.3}$$

if and only if there is a $(0,1)$-valued solution $U = [u_1, u_2, \ldots, u_n]^T$ to the matrix equation $AU = dB$, for some integer d.

Recall that the *Euclidean length* of a vector $\vec{y} \in \mathbb{R}^k$ is

$$||\vec{y}|| = \sqrt{y_1^2 + y_2^2 + \cdots + y_k^2}.$$

Thus, if $\vec{y} \in \mathcal{L}$ has the form of Equation 8.3, then the Euclidean length of $\vec{y}$ is

$$||\vec{y}|| = \sqrt{u_1^2 + u_2^2 + \cdots + u_n^2} \leq \sqrt{n},$$

because $u_i \in \{0,1\}$ for $i = 1, 2, \ldots, n$. Thus $\vec{y}$ is a vector whose Euclidean length is small when compared to most of the vectors in the lattice $\mathcal{L}$.

This leads us to consider the following optimization problem, Problem 8.1.

Problem 8.1:	Shortest Vector
Instance:	a matrix M with integer entries.
Find:	the minimum value of $\|\|\vec{y}\|\|$
	subject to $\vec{y} \in \mathcal{L}$, the lattice with basis M.

8.2 Theoretical development

Throughout this section, let

$$M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$$

be an m by n matrix considered as a set of column vectors in $\mathbb{Z}^m$, and let

$$\mathcal{L} = \mathsf{Span}_{\mathbb{Z}}(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n) = \{y : y = Mx, x \in \mathbb{Z}^n\}$$

be the lattice with basis M. We will describe an algorithm that will find a new basis M' for $\mathcal{L}$ containing vectors that have Euclidean length smaller than the vectors in those in M. The goal is to produce such a basis containing a vector $\vec{y}$ that solves Problem 8.1. First, in order to facilitate the analysis and development of the algorithm, we review some concepts from linear algebra.

Recall that essentially the only way we can alter the appearance of a basis, without changing the vector subspace that it spans, is to either

(i) reorder the vectors in the basis, or

(ii) perform an operation of the form:

$$\text{replace } \vec{b}_j \text{ with } \alpha_1\vec{b}_1 + \alpha_2\vec{b}_2 + \cdots + \alpha_n\vec{b}_n$$

where $\alpha_i \in \mathbb{R}$ for all i and $\alpha_j \neq 0$. However, we wish to have the new basis still be a basis for the lattice $\mathcal{L}$. Thus, we also require that $\alpha_i \in \mathbb{Z}$ for all i and that $\alpha_j = \pm 1$. Consequently, we can reduce any operation of type (ii) to performing a sequence of the following three operations:

1. Replace $\vec{b}_i$ with $\vec{b}_i + \vec{b}_j$
2. Replace $\vec{b}_i$ with $-\vec{b}_i + \vec{b}_j$
3. Replace $\vec{b}_i$ with $\vec{b}_i - \vec{b}_j$

For example, to obtain the operation,

$$\text{replace } \vec{b}_2 \text{ by } 2\vec{b}_1 + \vec{b}_2 - 3\vec{b}_3,$$

the following sequence of operations can be performed:

$$\begin{aligned}
&\text{replace } \vec{b}_2 \text{ with } b_2 - b_3\\
&\text{replace } \vec{b}_2 \text{ with } b_2 - b_3\\
&\text{replace } \vec{b}_2 \text{ with } b_2 - b_3\\
&\text{replace } \vec{b}_2 \text{ with } b_1 + b_2\\
&\textbf{replace } \vec{b}_2 \textbf{ with } b_1 + b_2.
\end{aligned}$$

If $\vec{x} = [x_1, x_2, \ldots, x_m]$ and $\vec{y} = [y_1, y_2, \ldots, y_m]$ are vectors in $\mathbb{R}^m$, then we denote by $\vec{x} \cdot \vec{y}$ the *dot product* of $\vec{x}$ and $\vec{y}$, defined as

$$\vec{x} \cdot \vec{y} = x_1 y_1 + x_2 y_2 + \cdots + x_m y_m.$$

We say that $\vec{x}$ and $\vec{y}$ are *orthogonal* if $\vec{x} \cdot \vec{y} = 0$. If the vectors in a basis are pairwise orthogonal, we say that it is an *orthogonal basis*. Notice that $||\vec{x}||$, the *Euclidean length* of $\vec{x}$, is $\sqrt{\vec{x} \cdot \vec{x}}$. The *triangle inequality* states that the sum of the lengths of two sides of a triangle is always greater than the length of the third. In terms of vectors $\vec{x}$ and $\vec{y}$, this says that

$$||\vec{x}|| + ||\vec{y}|| \geq ||\vec{x} + \vec{y}||.$$

Observe that, if the basis vectors $\vec{b}_i$ and $\vec{b}_j$ are orthogonal, then by the triangle inequality, we have

$$\begin{aligned} ||\vec{b}_i + \vec{b}_j|| &\geq \max\{||\vec{b}_i||, ||\vec{b}_j||\}, \\ ||-\vec{b}_i + \vec{b}_j|| &\geq \max\{||\vec{b}_i||, ||\vec{b}_j||\}, \\ &\text{and} \\ ||\vec{b}_i - \vec{b}_j|| &\geq \max\{||\vec{b}_i||, ||\vec{b}_j||\}. \end{aligned}$$

Hence, none of the three replacement operations can reduce the length of $\vec{b}_i$. Moreover, if all the vectors in the basis are pairwise orthogonal, then there is no way to further reduce the size of the vectors in the basis. Thus, in order to achieve a basis with vectors of minimal length, we will try to make a new basis for the lattice $\mathcal{L}$ in which the vectors in the basis are as pairwise orthogonal as possible. In general, we will not be able to fully orthogonalize M because we can only do the above type (ii) operations with integer coefficients. If real coefficients were allowed, then Algorithm 8.1 would produce a basis in which the vectors are pairwise orthogonal. Algorithm 8.1, the standard *Gram-Schmidt process* of orthogonalization from linear algebra, uses $O(n^3)$ arithmetic operations.

Algorithm 8.1: GRAM-SCHMIDT $(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n)$

$\vec{b}_1^* \leftarrow \vec{b}_1$
for $j \leftarrow 2$ **to** n
do $\begin{cases} \vec{b}_j^* \leftarrow \vec{b}_j \\ \textbf{for } i \leftarrow 1 \textbf{ to } j-1 \\ \quad \textbf{do} \begin{cases} \alpha_{ij} \leftarrow (\vec{b}_i^* \cdot \vec{b}_j)/||\vec{b}_i^*||^2 \\ \vec{b}_j^* \leftarrow \vec{b}_j^* - \alpha_{ij}\vec{b}_i^* \end{cases} \end{cases}$
return $([\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*], \{\alpha_{ij}\}_{i<j})$

A useful tool for measuring the volume of the parallelepiped determined by a set of vectors is the determinant.

Definition 8.1: The *sign* of a permutation on a set $\mathcal{X} = \{1, 2, \ldots, n\}$ is given by

$$\text{sign}(\sigma) = \begin{cases} +1 \text{ if } \sigma \text{ is an even permutation;} \\ -1 \text{ if } \sigma \text{ is an odd permutation.} \end{cases}$$

The *determinant* of the n by n matrix M is

$$\det M = \sum_{\sigma \in \text{Sym}(\mathcal{X})} \text{sign}(\sigma) \prod_{i=1}^{n} M[i, \sigma(i)].$$

Recall that Algorithm 2.19 can be used to compute the sign of a permutation. From Definition 8.1, it is straightforward to see that the determinant of the 2 by 2 matrix M is

$$\det M = M[1,1]M[2,2] - M[1,2]M[2,1]$$

because the only permutations of $\{1, 2\}$ are **I** (the identity permutation) and $(1, 2)$. Similarly, the determinant of the 3 by 3 matrix M is

$$\begin{aligned} \det M = {} & M[1,1]M[2,2]M[3,3] - M[1,1]M[2,3]M[3,2] \\ & - M[1,2]M[2,1]M[3,3] + M[1,2]M[2,3]M[3,1] \\ & + M[1,3]M[2,1]M[3,2] - M[1,3]M[2,2]M[3,1], \end{aligned}$$

because the permutations of $\{1, 2, 3\}$ are **I**, $(2, 3)$, $(1, 2)$, $(1, 2, 3)$, $(1, 3, 2)$ and $(1, 3)$. Furthermore, if we denote by $M_{\langle ij \rangle}$ the matrix obtained by removing row i and column j, then it is easily seen that we can recursively compute the determinant of M with the formula:

$$\det M = \sum_{j=1}^{n} (-1)^{i+j} M[i, j] \det M_{\langle ij \rangle},$$

where i is any fixed row. Similarly, the determinant can be computed with the formula

$$\det M = \sum_{i=1}^{n} (-1)^{i+j} M[i, j] \det M_{\langle ij \rangle}$$

where j is any fixed column. This method of computing the determinant is called *cofactor expansion* or *Laplace expansion* and will take $O(n!)$ operations. A method that uses only $O(n^3)$ operations is to use elementary row operations to reduce the matrix to upper triangular form; see any text on linear algebra.

Elementary linear algebra establishes Lemma 8.3.

LEMMA 8.3 *Let $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_m \in \mathbb{R}^m$ be linearly independent.*

(i) *On input $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$, Algorithm 8.1 computes an orthogonal basis*

$$M^* = [\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*]$$

for the vector space $\mathsf{Span}_{\mathbb{R}}(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n)$.

(ii) *$\vec{b}_k^*$ is orthogonal to each vector in*

$$\mathsf{Span}_{\mathbb{R}}(\vec{b}_1^*, \ldots, \vec{b}_{k-1}^*).$$

(iii) *If M is a square matrix, then*

$$|\det M| = |\det M^*| = \prod_{j=1}^{n} ||\vec{b}_j^*||.$$

In view of Lemma 8.3, we define, for any lattice $\mathcal{L}$, with (possibly non-square) linearly independent basis matrix $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$, the quantity

$$\mathsf{vol}(\mathcal{L}) = \prod_{j=1}^{n} ||\vec{b}_i^*||.$$

$\mathsf{vol}(\mathcal{L})$ is the n-dimensional *volume* of the parallelepiped with vertices

$$\left\{ \sum_{j=1}^{n} \vec{b}_j^* x_j \ : \quad x_j \in \{0,1\} \quad \forall j \right\},$$

where $[\vec{b}_1^*, \ldots, \vec{b}_n^*]$ is the orthogonal basis obtained by applying Algorithm 8.1. This volume is independent of the choice of linearly independent basis for $\mathcal{L}$.

If M is a basis for a lattice $\mathcal{L}$, we define the *weight* of M to be

$$\mathsf{wt}(M) = \prod_{j=1}^{n} ||\vec{b}_i||.$$

The relationship between $\mathsf{wt}(M)$ and $\mathsf{vol}(\mathcal{L})$ is provided by *Hadamard's inequality*, which we state now.

LEMMA 8.4 *(Hadamard's inequality) For all bases M of $\mathcal{L}$, we have*

$$\mathsf{wt}(M) \geq \mathsf{vol}(\mathcal{L}).$$

PROOF Suppose $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ is a basis for the lattice $\mathcal{L}$. Then on input M, Algorithm 8.1 constructs, for each $j = 1, 2, \ldots, n$, the vectors

$$\vec{b}_j^* = \vec{b}_j - \sum_{i=1}^{j-1} \alpha_{ij} \vec{b}_i^*,$$

where

$$\alpha_{ij} = \frac{(\vec{b}_i^* \cdot \vec{b}_j)}{\|\vec{b}_i^*\|^2}.$$

Thus,

$$\vec{b}_j^* \cdot \vec{b}_j^* = \vec{b}_j^* \cdot \left(\vec{b}_j - \sum_{i=1}^{j-1} \alpha_{ij} \vec{b}_i^* \right) = \vec{b}_j^* \cdot \vec{b}_j,$$

because $\{\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*\}$ is an orthogonal basis. If θ is the angle between $\vec{b}_j$ and $\vec{b}_j^*$, then

$$\begin{aligned} \|\vec{b}_j^*\|^2 &= \vec{b}_j^* \cdot \vec{b}_j^* \\ &= \vec{b}_j^* \cdot \vec{b}_j \\ &= \|\vec{b}_j\| \cdot \|\vec{b}_j^*\| \cos(\theta). \end{aligned}$$

Thus $0 \leq \cos(\theta) \leq 1$, and so for all j we have $\|\vec{b}_j^*\| \leq \|\vec{b}_j\|$. The result now follows. ∎

Hadamard's inequality is illustrated in Example 8.2.

Example 8.2

Let

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 & 3 \\ 1 & 0 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 3 \\ 1 & 1 & -1 & 0 & 0 \end{bmatrix}.$$

To obtain the determinant of M, we use a cofactor expansion along column 5.

$$\begin{aligned} \det M &= 3 \cdot \det \begin{bmatrix} 1 & 0 & 2 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ 1 & 1 & -1 & 0 \end{bmatrix} + (-3) \det \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 2 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & -1 & 0 \end{bmatrix} \\ &= 3(-1) \begin{bmatrix} 1 & 0 & 1 \\ 0 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix} - 3 \left(-1 \cdot \det \begin{bmatrix} 1 & 0 & 2 \\ 1 & 0 & 1 \\ 1 & 1 & -1 \end{bmatrix} + 1 \cdot \det \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & -1 \end{bmatrix} \right) \\ &= 3(-1) \cdot (1) - 3(-1 \cdot 1 + 1 \cdot (-1 + 2)) \\ &= -3. \end{aligned}$$

The weight of M is

$$\mathrm{wt}(M) = \sqrt{4} \cdot \sqrt{3} \cdot \sqrt{7} \cdot \sqrt{2} \cdot \sqrt{18} = 12\sqrt{21} \approx 54.990908.$$

Now applying Algorithm 8.1 we obtain:

$$\alpha = \begin{bmatrix} * & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} \\ * & * & -\frac{3}{2} & 0 & -\frac{3}{4} \\ * & * & * & \frac{2}{3} & -\frac{1}{2} \\ * & * & * & * & 6 \\ * & * & * & * & * \end{bmatrix}$$

and

$$M^* = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{4} & \frac{1}{3} & \frac{3}{4} \\ 1 & -\frac{1}{2} & \frac{3}{4} & 0 & -\frac{3}{4} \\ 1 & -\frac{1}{2} & -\frac{1}{4} & -\frac{1}{3} & \frac{3}{4} \\ 0 & -1 & -\frac{1}{2} & \frac{1}{3} & 0 \\ 1 & \frac{1}{2} & -\frac{3}{4} & 0 & -\frac{3}{4} \end{bmatrix}$$

(Recall that we have defined α_{ij} only for $i < j$.) Observe that

$$\begin{aligned} \mathrm{vol}(\mathcal{L}) &= \|\vec{b}_1^*\| \cdot \|\vec{b}_2^*\| \cdot \|\vec{b}_3^*\| \cdot \|\vec{b}_4^*\| \cdot \|\vec{b}_5^*\| \\ &= \sqrt{4} \cdot \sqrt{2} \cdot \sqrt{\frac{6}{4}} \cdot \sqrt{\frac{1}{3}} \cdot \sqrt{\frac{9}{4}} \\ &= 3 \\ &= |\det M|. \end{aligned}$$

□

The key to developing an algorithm to find the shortest vector in a lattice is revealed in Lemma 8.6. First, recall Cramer's rule, which is easily established using elementary linear algebra.

LEMMA 8.5 *(Cramer's rule) Let $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ be an invertible n by n matrix with entries in $\mathbb{R}$. Then, for any vector $\vec{y} \in \mathbb{R}^n$, the i-th component of the solution $\vec{x}$ to the matrix equation $\vec{y} = M\vec{x}$ is*

$$x_j = \frac{\det M_j}{\det M},$$

where $M_j = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_{j-1}, \vec{y}, \vec{b}_{j+1}, \ldots, \vec{b}_n]$.

LEMMA 8.6 *Given a full dimensional lattice $\mathcal{L}$, with basis M, let $\vec{y} = M\vec{x}$ be a shortest vector in $\mathcal{L}$. Then for all $i = 1, 2, \ldots, n$, the i-th component of $\vec{x}$ satisfies*

$$|x_i| \leq \frac{\mathrm{wt}(M)}{\mathrm{vol}(\mathcal{L})}.$$

PROOF Using Lemma 8.5, we have

$$|x_i| = \frac{|\det M_j|}{|\det M|},$$

where $M_j = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_{j-1}, \vec{y}, \vec{b}_{j+1}, \ldots, \vec{b}_n]$. By Lemma 8.4, we obtain

$$|\det M_j| \leq ||\vec{b}_1|| \cdot ||\vec{b}_2|| \cdots ||\vec{b}_{j-1}|| \cdot ||\vec{y}|| \cdot ||\vec{b}_{j+1}|| \cdots ||\vec{b}_n||.$$

Also $\vec{b}_j \in \mathcal{L}$, and so $||\vec{y}|| \leq ||\vec{b}_j||$. Thus

$$|\det M_j| \leq ||\vec{b}_1|| \cdot ||\vec{b}_2|| \cdots ||\vec{b}_{j-1}|| \cdot ||\vec{b}_j|| \cdot ||\vec{b}_{j+1}|| \cdots ||\vec{b}_n|| = \mathrm{wt}(M).$$

Therefore $|x_i| \leq \mathrm{wt}(M)/\mathrm{vol}(\mathcal{L})$ as claimed, because $\mathrm{vol}(\mathcal{L}) = |\det M|$ when $\mathcal{L}$ is full dimensional. ∎

As a consequence of Lemma 8.6, we are motivated to find a basis M' such that the ratio

$$\frac{\mathrm{wt}(M')}{\mathrm{vol}(\mathcal{L})}$$

is small. Thus, because $\mathrm{vol}(\mathcal{L})$ depends only on the lattice $\mathcal{L}$ and not on the basis, we search for a basis M' with minimal weight $\mathrm{wt}(M')$. We have the following lower bound on the shortest vector in $\mathcal{L}$.

LEMMA 8.7 *Let $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ be a basis for the lattice $\mathcal{L}$. If $\vec{b} \in \mathcal{L}$, $b \neq 0$, then*

$$||\vec{b}|| \geq \min_j ||\vec{b}_j^*||,$$

where $[\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*]$ is the result of running Algorithm 8.1 on input M.

PROOF Let $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ be a basis for the lattice $\mathcal{L}$. Then $b \in \mathcal{L}$ implies that there is a $k \leq n$ such that

$$\vec{b} = \sum_{j=1}^{k} \vec{b}_j z_j$$

with $z_j \in \mathbb{Z}$, $z_k \neq 0$. Algorithm 8.1 sets

$$\vec{b}_j^* = \vec{b}_j - \sum_{i=1}^{j-1} \alpha_{ij} \vec{b}_i^*,$$

where

$$\alpha_{ij} = \frac{(\vec{b}_i^* \cdot \vec{b}_j)}{||\vec{b}_i^*||^2}.$$

Thus, we see that

$$\vec{b} = \sum_{j=1}^{k} \vec{b}_j^* z_j^*,$$

where $z_j^* \in \mathbb{R}$ for $j = 1, 2, \ldots, k-1$ and $z_k^* = z_k \in \mathbb{Z}$. Now, because $\{\vec{b}_1^*, \vec{b}_2^*, \cdots, \vec{b}_n^*\}$ is an orthogonal basis, we have

$$\begin{aligned} ||\vec{b}|| &= \left(\sum_{j=1}^{k} (\vec{b}_j^* \cdot \vec{b}_j^*)(z_j^*)^2 \right)^{1/2} \\ &\geq |z_k| \cdot ||\vec{b}_k^*|| \\ &\geq \min_j ||\vec{b}_j^*||. \end{aligned}$$

∎

The basis $M^* = [\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*]$ computed by Algorithm 8.1 on input M is typically not a basis for the lattice with basis M. This is because the coefficients $\{\alpha_{i,j}\}_{i<j}$ are not necessarily integers. Lemma 8.7 suggests that we should look for a basis that is as close to orthogonal as we can find.

Definition 8.2: Let $\mathcal{L}$ be a lattice with basis $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ and let $M^* = [\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*]$ be the basis for $\mathrm{Span}_{\mathbb{R}}(M)$ obtained by Algorithm 8.1 on input M. We say that M is a *reduced basis* if:

(a) $|\alpha_{i,j}| \leq \frac{1}{2}$ for all $i < j$; and

(b) $||\vec{b}_{j+1}^* + \alpha_{j,j+1}\vec{b}_j^*||^2 \geq \frac{3}{4}||\vec{b}_j^*||^2$ for all $j = 1, 2, \ldots, n-1$.

We are interested in this definition because we can establish the following result and obtain an upper bound on the length of the shortest vector in a lattice $\mathcal{L}$.

THEOREM 8.8 *Let $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ be a reduced basis for the lattice $\mathcal{L}$. Then the following hold:*

1. $||\vec{b}_1|| \leq 2^{(n-1)/4}\mathrm{vol}(\mathcal{L})^{1/n}$; *and*
2. $\mathrm{wt}(M) \leq 2^{n(n-1)/4}\mathrm{vol}(\mathcal{L})$.

PROOF Suppose $[\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ is a reduced basis for the lattice $\mathcal{L}$. Then the orthogonality of $\vec{b}_j^*$ and $\vec{b}_{j+1}^*$ implies, by part (b) of Definition 8.2, that

$$\frac{3}{4}||\vec{b}_j^*||^2 \leq ||\vec{b}_{j+1}^* + \alpha_{j,j+1}\vec{b}_j^*||^2 = ||\vec{b}_{j+1}^*||^2 + \alpha_{j,j+1}^2||\vec{b}_j^*||^2.$$

Now, applying part (a), we have $\|\vec{b}_{j+1}^*\|^2 \geq \frac{1}{2}\|\vec{b}_j^*\|^2$. Iterating this inequality, it follows that

$$\|\vec{b}_j^*\|^2 \geq \left(\frac{1}{2}\right)^{j-1} \|\vec{b}_1^*\|^2. \tag{8.4}$$

Since $\vec{b}_1^* = \vec{b}_1$ we obtain

$$\begin{aligned}(\mathrm{vol}(\mathcal{L}))^2 &= \prod_{j=1}^{n} \|\vec{b}_j^*\|^2 \\ &\geq \left(\frac{1}{2}\right)^{\sum_{j=1}^{n}(j-1)} \|\vec{b}_1\|^{2n} \\ &= \left(\frac{1}{2}\right)^{n(n-1)/2} \|\vec{b}_1\|^{2n}.\end{aligned}$$

This yields part 1.

For part 2, first recall that Algorithm 8.1 sets

$$\vec{b}_j^* = \vec{b}_j - \sum_{i=1}^{j-1} \alpha_{ij} \vec{b}_i^*,$$

where

$$\alpha_{ij} = \frac{(\vec{b}_i^* \cdot \vec{b}_j)}{\|\vec{b}_i^*\|^2}.$$

Thus,

$$\vec{b}_j = \sum_{i=1}^{j} \alpha_{ij} \vec{b}_i^*,$$

with $\alpha_{jj} = 1$. Furthermore, because the vectors $\{\vec{b}_i^*\}$ are orthogonal it follows that

$$\|\vec{b}_j\|^2 = \sum_{i=1}^{j} \alpha_{ij}^2 \|\vec{b}_i^*\|^2 \leq \|\vec{b}_j^*\|^2 + \frac{1}{4} \sum_{i=1}^{j-1} \|\vec{b}_i^*\|^2$$

by part (a) of Definition 8.2. Now, using Equation 8.4, we have

$$\|\vec{b}_i^*\|^2 \leq 2^{j-i} \|\vec{b}_j^*\|^2,$$

and hence

$$\|\vec{b}_j\|^2 \leq \|\vec{b}_j^*\|^2 \left(1 + \frac{1}{4} \sum_{i=1}^{j-1} 2^{j-i}\right) \leq 2^{j-1} \|\vec{b}_j^*\|^2.$$

Therefore,

$$\begin{aligned}(\mathrm{wt}(M))^2 &= \prod_{j=1}^{n} \|\vec{b}_j\|^2 \\ &\leq 2^{n(n-1)/2} \prod_{j=1}^{n} \|\vec{b}_j^*\|^2 \\ &= 2^{n(n-1)/2}(\mathrm{vol}(\mathcal{L}))^2,\end{aligned}$$

as claimed. ∎

Example 8.3 *Continuation of Example 8.2*
The following computations show that

$$M' = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 \\ 0 & -1 & 0 & -1 & 1 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & -1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} M$$

where M is the matrix given in Example 8.2, and that M' is a reduced basis for a lattice $\mathcal{L}$. From Algorithm 8.1 we have:

$$\alpha' = \begin{bmatrix} * & 0 & 0 & 0 & 0 \\ * & * & 0 & \frac{1}{2} & 0 \\ * & * & * & -\frac{1}{2} & -\frac{1}{2} \\ * & * & * & * & -\frac{1}{2} \\ * & * & * & * & * \end{bmatrix}$$

and

$$M'^* = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} & \frac{3}{4} \\ 0 & -1 & 0 & \frac{1}{2} & -\frac{3}{4} \\ 0 & -1 & 0 & -\frac{1}{2} & \frac{3}{4} \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -\frac{1}{2} & -\frac{3}{4} \end{bmatrix}$$

Hence part (a) of Definition 8.2 is satisfied. To check part (b) we must show that

$$\|\vec{b}'^*_{j+1} + \alpha'_{j,j+1}\vec{b}'^*_j\|^2 \geq \frac{3}{4}\|\vec{b}^*_j\|^2$$

for each $j = 1, 2, 3, 4$. To see this, observe that for $j = 1, 2, 3, 4$ we have

$$\|\vec{b}'^*_{j+1} + \alpha'_{j,j+1}\vec{b}'^*_j\|^2 = 2, 2, \frac{3}{2}, \frac{5}{2},$$

and

$$\frac{3}{4}\|\vec{b}_j^*\|^2 = \frac{3}{4}, \frac{3}{2}, \frac{3}{2}, \frac{3}{4},$$

respectively. Thus, the basis satisfies part (b) and consequently M' is a reduced basis for $\mathcal{L} = \mathcal{L}(M)$. Checking the bounds in Theorem 8.8, we observe that

$$\begin{aligned}\|\vec{b}_1'\| &\leq 2^1(\mathrm{vol}(\mathcal{L}))^{1/5} \\ &= 2 \cdot 3^{\frac{1}{5}} \\ &\approx 2.4914618;\end{aligned}$$

and

$$\begin{aligned}\mathrm{wt}(M') = \prod_{j=1}^{5} \|\vec{b}_j'\| &= 2(6)^{\frac{1}{2}} \approx 4.898979 \\ &\leq 2^5 \mathrm{vol}(\mathcal{L}) = 64 \cdot 3^{\frac{1}{5}} \approx 79.7267801.\end{aligned}$$

□

8.3 A reduced basis algorithm

Theorem 8.8 shows that, if we have a reduced basis for a lattice $\mathcal{L}$, then it contains a vector that has length less than or equal to $2^{(n-1)/4}\mathrm{vol}(\mathcal{L})^{1/n}$. This is a good start towards finding a shortest vector in $\mathcal{L}$. In this section we present Algorithm 8.2 which computes a reduced basis for $\mathcal{L}$.

Algorithm 8.2: LLL $(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n)$

external GRAM-SCHMIDT()
done $\leftarrow$ **false**
$([\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*], \{\alpha_{ij}\}_{i<j}) \leftarrow$ GRAM-SCHMIDT$(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n)$
while not *done*
do
  for $j \leftarrow 2$ **to** n
    do
      for $i \leftarrow j-1$ **downto** 1
        do if $|\alpha_{ij}| > \frac{1}{2}$
        then $\vec{b}_j \leftarrow \vec{b}_j - \lfloor \alpha_{ij} + \frac{1}{2} \rfloor \vec{b}_i$
  if $\|\vec{b}_{j+1}^* + \alpha_{j,j+1}\vec{b}_j^*\|^2 < \frac{3}{4}\|\vec{b}_j^*\|^2$ for some j
    then interchange $\vec{b}_j$ and $\vec{b}_{j+1}$
    else *done* $\leftarrow$ **true**
  $([\vec{b}_1^*, \vec{b}_2^*, \ldots, \vec{b}_n^*], \{\alpha_{ij}\}_{i<j}) \leftarrow$ GRAM-SCHMIDT$(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n)$
return $([\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n])$

THEOREM 8.9 *Let $\mathcal{L}$ be a lattice with basis $[\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$, and let*

$$Max = \max\{||b_i||^2 : i = 1, 2, \ldots, n\} \geq 2.$$

Then Algorithm 8.2 finds a reduced basis for $\mathcal{L}$ using at most $O(n^5 \log(Max))$ arithmetic operations.

PROOF When $|\alpha_{ij}| > \frac{1}{2}$, Algorithm 8.2 replaces $\vec{b}_j$ with

$$\vec{x} = \vec{b}_j - \left\lfloor \alpha_{ij} + \frac{1}{2} \right\rfloor \vec{b}_i.$$

Now

$$\begin{aligned}
\left| \frac{(\vec{b}_i^* \cdot \vec{x})}{||\vec{b}_i^*||^2} \right| &= \left| \frac{\vec{b}_i^* \cdot \left(\vec{b}_j - \lfloor \alpha_{ij} + \frac{1}{2} \rfloor \vec{b}_i \right)}{||\vec{b}_i^*||^2} \right| \\
&= \left| \frac{(\vec{b}_i^* \cdot \vec{b}_j)}{||\vec{b}_i^*||^2} - \left\lfloor \alpha_{ij} + \frac{1}{2} \right\rfloor \frac{(\vec{b}_i^* \cdot \vec{b}_i)}{||\vec{b}_i^*||^2} \right| \\
&= \left| \alpha_{ij} - \left\lfloor \alpha_{ij} + \frac{1}{2} \right\rfloor \right| \\
&\leq \frac{1}{2}.
\end{aligned}$$

Thus when Algorithm 8.1 is called at the end of the loop, the recalculated α_{ij}s will satisfy part (a) of Definition 8.2. It is also easy to see that, if the algorithm terminates, then condition (b) of Definition 8.2 will be met. To see that the algorithm terminates, we define for $\ell = 0, 1, 2, \ldots, n$, the quantity

$$v_\ell = \prod_{j=1}^{\ell} ||\vec{b}_j^*||^2.$$

Notice that each $v_i > 0$. In particular $v_0 = 1$ and $v_n = \text{vol}(\mathcal{L})^2$. Furthermore, elementary linear algebra shows that v_ℓ is the determinant of the ℓ by ℓ matrix whose $[i, j]$ entry is $\vec{b}_i \cdot \vec{b}_j$. In particular, because the $\vec{b}_j$s are linearly independent vectors with integer entries, we have that $v_\ell > 1$. Thus

$$V = \prod_{\ell=1}^{n-1} v_\ell > 1.$$

Furthermore, the value of V changes only when some $\vec{b}_j$ is changed, and this occurs only if

$$||\vec{b}_{j+1}^* + \alpha_{j,j+1}\vec{b}_j^*||^2 < \frac{3}{4}||\vec{b}_j^*||^2,$$

for some j. In this case, $\vec{b}_j$ and $\vec{b}_{j+1}$ are interchanged, and Algorithm 8.1 recalculates the corresponding orthogonal basis and α_{ij}s. Let $\widehat{b}_1, \widehat{b}_2, \ldots, \widehat{b}_n$ be the new basis after the interchange, and let $\widehat{b}_1^*, \widehat{b}_2^*, \ldots, \widehat{b}_n^*$ be the recomputed orthogonal basis. Then

$$\begin{aligned} \widehat{b}_i &= \vec{b}_i \text{ when } i \neq j \text{ or } j+1; \\ \widehat{b}_i^* &= \vec{b}_i^* \text{ when } i \neq j \text{ or } j+1; \\ \widehat{b}_j &= \vec{b}_{j+1}; \text{ and} \\ \widehat{b}_{j+1} &= \vec{b}_j. \end{aligned}$$

After the call to Algorithm 8.1, we have

$$\widehat{b}_j^* = \widehat{b}_j - \sum_{i=1}^{j-1} \widehat{\alpha}_{ij} \widehat{b}_i^*, \tag{8.5}$$

where

$$\widehat{\alpha}_{ij} = \frac{\widehat{b}_i^* \cdot \widehat{b}_j}{||\widehat{b}_i^*||^2}$$

and $|\widehat{\alpha}_{ij}| \leq \frac{1}{2}$. On the other hand, we have

$$b_{j+1}^* = b_{j+1} - \sum_{i=1}^{j} \alpha_{ij+1} b_i^*, \tag{8.6}$$

$\widehat{b}_j = \vec{b}_{j+1}$, and $\alpha_{i,j+1} = \widehat{\alpha}_{i,j}$ for $i = 1, 2, \ldots, j-1$. Thus, substituting into Equation 8.5, we see that

$$\widehat{b}_j^* = \alpha_{j,j+1} \vec{b}_j^* + \vec{b}_{j+1}^*.$$

Therefore,

$$||\widehat{b}_j^*||^2 = ||\alpha_{j,j+1} \vec{b}_j^* + \vec{b}_{j+1}^*||^2 < \frac{3}{4} ||\vec{b}_j^*||^2.$$

Thus when $\vec{b}_j$ and $\vec{b}_{j+1}$ are interchanged, the number v_j is reduced by a factor of $3/4$ (or more). The other v_i are unchanged and so each time there is an interchange, V is reduced by a factor of $3/4$ (or more). Therefore the algorithm terminates, because throughout Algorithm 8.2 we always have $V > 1$. The initial value of v_i is by Lemma 8.4 at most $(Max)^i$. It follows that, prior to the execution of the **while** loop, V is at most $(Max)^{\frac{n(n-1)}{2}}$. Consequently, if t is the number of interchanges performed, then

$$t \leq \log_{\frac{3}{4}}((Max)^{n(n-1)/2}) = \frac{n(n-1)}{2} \log_{\frac{3}{4}}(Max).$$

Thus t is $O(n^2 \log(Max))$, and because Algorithm 8.1 is an $O(n^3)$ algorithm, it follows that Algorithm 8.2 uses $O(n^5 \log(Max))$ arithmetic operations.

∎

8.4 Solving systems of integer equations

In this section, we return to the problem of solving a matrix equation of the form $AU = B$ for a $(0,1)$-valued vector U, where A is an m by n integer valued matrix and $B \in \mathbb{Z}^m$. As outlined in Section 8.1, we set M equal to the following $m+n$ by $n+1$ matrix:

$$M = \begin{bmatrix} I & \vec{0} \\ A & -B \end{bmatrix},$$

and consider the lattice $\mathcal{L}$ with basis M.

The basic idea of the algorithm is that if U is very short, then there is a good chance that it will appear in a reduced basis M' for $\mathcal{L}$. Thus we check whether the reduced basis contains a vector of the form $[U, \vec{0}]$ with $U \in \{0,1\}^n$. If it does, then a solution to $AU = dB$ is found for some integer d; otherwise another approach must be taken. We can also check if the reduced basis contains a vector $[U, \vec{0}]$ with $U \in \{0,-1\}^n$, for then $-U \in \{0,1\}^n$ and $A(-U) = dB$.

Example 8.4 *Using Algorithm 8.2 to find a solution.*

Consider the incidence matrix

$$A = \begin{bmatrix} 1 & 1 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 \\ 1 & 3 & 6 & 6 & 3 & 1 & 3 & 3 & 6 & 3 \end{bmatrix}$$

of Example 6.12.1, in which we have added the orbit length equation in the last row, as suggested by Exercise 6.17. For this matrix, we wish to solve $AU = B$ where

$$B = [1,1,1,1,1,1,7]^{\mathsf{T}}.$$

The basis M is as follows:

$$M = \left[\begin{array}{cccccccccc|c}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
\hline
1 & 1 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\
0 & 0 & 2 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & -1 \\
0 & 0 & 0 & 2 & 1 & 1 & 0 & 0 & 0 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 2 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & -1 \\
1 & 3 & 6 & 6 & 3 & 1 & 3 & 3 & 6 & 3 & -7
\end{array}\right]$$

The weight of this basis is $\mathrm{wt}(M) \approx 4504883.126564$. Applying Algorithm 8.2 to M, we obtain the reduced basis

$$M' = \left[\begin{array}{cccccccccc|c}
1 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & -1 & -1 \\
0 & 0 & -1 & 0 & 1 & 0 & 1 & -1 & -1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
1 & 0 & -1 & 0 & 0 & -1 & 1 & -2 & -1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 1 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 \\
1 & 1 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 1
\end{array}\right]$$

The weight of the new basis is $\mathrm{wt}(M') \approx 10571.993190$, which is, as expected, less than $\mathrm{wt}(M)$. Also, column 4 of M' is of the form

$$[u_1, u_2, \ldots, u_{10}, 0, 0, 0, 0, 0, 0, 0]^{\mathsf{T}}$$

with each $u_i \in \{0, 1\}$. This gives the solution

$$U = [u_1, u_2, \ldots, u_{10}]^T = [1, 0, 0, 0, 1, 0, 0, 1, 0, 0]^T$$

to $AU = dB$ with $d = 1$.

□

In Example 8.4, Algorithm 8.2 succeeds in finding a solution to $AU = dB$. However, there are many situations in which it will not succeed. This is because the weight of reduced basis obtained by Algorithm 8.2 may be still too large to guarantee that the basis contains the shortest vectors in the lattice (see Example 8.5). Therefore, we will now consider another method to reduce the weight of the basis. Suppose the current basis is $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ and consider combinations of the form

$$\vec{v} = \vec{b}_i + \epsilon \vec{b}_j$$

where $\epsilon = \pm 1$. If $||v|| < \max\{||\vec{b}_i||, ||\vec{b}_j||\}$, then one of $||\vec{b}_i||$ and $||\vec{b}_j||$ can be replaced by v and the weight of the basis will be reduced. In order to facilitate the implementation of this idea, we will use the array Δ defined by

$$\Delta_{i,j} = \vec{b}_i \cdot \vec{b}_j.$$

Let $k \in \{i, j\}$ be such that $||\vec{b}_k|| = \max\{||\vec{b}_i||, ||\vec{b}_j||\}$. When $\vec{b}_k$ is replaced by $\vec{v}$, the only entries of Δ that need to be recomputed are $\Delta_{h,k}$ and $\Delta_{k,h}$ for each h. Observe, for $h \neq k$, that

$$\begin{aligned}
\Delta_{k,h} &= \vec{v} \cdot \vec{b}_h \\
&= (\vec{b}_i + \epsilon \vec{b}_j) \cdot \vec{b}_h \\
&= (\vec{b}_i \cdot \vec{b}_h) + \epsilon (\vec{b}_j \cdot \vec{b}_h) \\
&= \Delta_{i,h} + \epsilon \Delta_{j,h},
\end{aligned}$$

and

$$\begin{aligned}
\Delta_{k,k} &= \vec{v} \cdot \vec{v} \\
&= (\vec{b}_i + \epsilon \vec{b}_j) \cdot (\vec{b}_i + \epsilon \vec{b}_j) \\
&= \Delta_{i,i} + \Delta_{j,j} + 2\epsilon \Delta_{i,j}.
\end{aligned}$$

Thus, Δ can be updated in $O(n)$ operations. This method of reducing the weight of the basis is simple to implement, and we present it as Algorithm 8.3. There are $\binom{n}{2}$ pairs of vectors to check and so this algorithm runs in $O(n^3)$ time.

Algorithm 8.3: WEIGHTREDUCTION $(\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n)$

global $\Delta_{ij}, 1 \leq i, j \leq n$
for $i \leftarrow 1$ **to** n
do {
  for $j \leftarrow i+1$ **to** n
  do {
    for each $\epsilon \in \{-1, 1\}$
    do {
      if $\Delta_{i,i} < \Delta_{j,j}$ **then** $k \leftarrow j$ **else** $k \leftarrow i$
      $\vec{v} \leftarrow \vec{b}_i + \epsilon \vec{b}_j$
      if $||\vec{v}||^2 < \Delta_{k,k}$
      then {
        $\Delta_{k,k} \leftarrow \Delta_{i,i} + \Delta_{j,j} + 2\epsilon \Delta_{i,j}$
        for $h \leftarrow 1$ **to** n
        do if $h \neq i$ **and** $h \neq j$
        then { $\Delta_{k,h} \leftarrow \Delta_{i,h} + \epsilon \Delta_{j,h}$; $\Delta_{h,k} \leftarrow \Delta_{k,h}$ }
        if $k \neq i$
        then { $\Delta_{k,i} \leftarrow \Delta_{i,i} + \epsilon \Delta_{j,i}$; $\Delta_{i,k} \leftarrow \Delta_{k,i}$ }
        else { $\Delta_{k,j} \leftarrow \Delta_{i,j} + \epsilon \Delta_{j,j}$; $\Delta_{j,k} \leftarrow \Delta_{k,j}$ }
        $\vec{b}_k \leftarrow \vec{v}$
      }
    }
  }
}
return $([\vec{b}_1, \vec{b}_2, \ldots, b_n])$

We see that progress can be made, using these algorithms, to reduce the weight of the basis and possibly converge to a solution. It turns out that using a combination of Algorithms 8.2 and 8.3 is often superior for reducing the weight than using either of them alone. Among the many possible ways to do this, a successful and simple method is given in Algorithm 8.4.

Algorithm 8.2 and Algorithm 8.1 work through the basis $[\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ from left to right. Thus the order in which the basis vectors appear has an effect on the outcome of Algorithm 8.2 and thus also on the outcome of Algorithm 8.4. For a given input basis, it may be prudent to order the basis vectors in increasing order of length, while for another basis we may prefer a decreasing order, and still another could do well with random ordering. Since the algorithms run relatively quickly, many such approaches can be tried. One can even try sorting the basis before each call to Algorithm 8.3 and also to include in the **while** loop additional calls to Algorithm 8.2. The optimal approach depends on the input basis and is best determined by experimentation.

Algorithm 8.4: KR $(M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n])$

external LLL(), WEIGHTREDUCTION()
Sort the basis vectors so that $||\vec{b}_1|| < ||\vec{b}_2|| < \cdots < ||\vec{b}_n||$
$M \leftarrow \text{LLL}(M)$
for $i \leftarrow 1$ **to** n
do { **for** $j \leftarrow 1$ **to** n **do** $\Delta_{i,j} \leftarrow \vec{b}_i \cdot \vec{b}_j$
$weight \leftarrow \prod_{i=1}^{n} \sqrt{\Delta_{i,i}}$
$done \leftarrow$ **false**
while not *done* { $M \leftarrow$ WEIGHTREDUCTION(M)
$new \leftarrow \prod_{i=1}^{n} \sqrt{\Delta_{i,i}}$
if $new < weight$ **then** $weight \leftarrow new$ **else** $done \leftarrow$ **true**
return (M)

Example 8.5

Consider the incidence matrix

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 \\ 1 & 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 0 & 2 & 0 & 2 & 0 \\ 0 & 0 & 1 & 1 & 2 & 1 & 1 & 0 & 2 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 2 & 2 & 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 1 \\ 15 & 15 & 15 & 15 & 30 & 30 & 30 & 5 & 30 & 15 & 30 & 30 & 30 & 30 & 30 & 30 & 30 & 30 & 15 \end{bmatrix}$$

of Example 6.12.2, in which we have added the orbit length equation in the last row, as suggested by Exercise 6.17. For this matrix, we wish to solve $AU = B$, where

$$B = [1, 1, 1, 1, 1, 35]^{\mathsf{T}}.$$

Applying Algorithm 8.2 to the basis

$$M = \begin{bmatrix} I & \vec{0} \\ A & -B \end{bmatrix},$$

we obtain the basis M' given below

$$\left[\begin{array}{cccccccccccccccccc|c}
0 & 0 & -1 & -1 & 0 & -1 & -1 & -1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & -3 & 3 & -2 \\
0 & 0 & 1 & 0 & -1 & 1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & 1 & 0 & 0 & 1 & 3 & -3 & 2 \\
-1 & 0 & 0 & 1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 \\
0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 2 & -3 & 2 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & -1 & 0 & -1 & 0 & -1 & 0 & -1 & 0 & 1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 \\
0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 2 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5
\end{array}\right]$$

The weight of this new basis is $\mathrm{wt}(M') \approx 13152322.331817$. Unfortunately there is no column of the form

$$[u_1, u_2, \ldots, u_{19}, 0, 0, 0, 0, 0, 0]^{\mathsf{T}}$$

with each $u_i \in \{0, 1\}$ or with each $u_i \in \{0, -1\}$. Applying Algorithm 8.4 to the

basis M', we obtain the basis M'', which is as follows:

$$\left[\begin{array}{cccccccccccccccccc|c}
0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & -1 & 0 \\
-1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \\
\hline
0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 \\
0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5
\end{array}\right]$$

The weight of this new basis is $\mathrm{wt}(M'') \approx 927342.111629$, and this is less than $\mathrm{wt}(M')$ for the previous basis. Furthermore, column 17 has the form

$$[u_1, u_2, \ldots, u_{19}, 0, 0, 0, 0, 0, 0]^{\mathsf{T}}$$

with each $u_i \in \{0, 1\}$. This gives the solution

$$\begin{aligned} U &= [u_1, u_2, \ldots, u_{19}]^{\mathsf{T}} \\ &= [0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1]^{\mathsf{T}} \end{aligned}$$

to $AU = dB$ with $d = 1$. □

8.5 The Merkle-Hellman knapsack system

A *public key cryptosystem* is a method of secure transmission of messages in which the sender looks up the receiver's key K from a publicly available list of keys $\mathcal{K}$. From the key, an encryption rule is determined, which is used to

encode *plaintext* messages $\mathcal{P}$ into *ciphertext* messages $\mathcal{C}$. A ciphertext message is then transmitted to the receiver. Only the receiver knows the decryption rule, corresponding to the *key* K, that will decode messages sent to him. Anyone else will have great difficulty in decrypting the ciphertext, even though they know the public key K and the encryption rule used.

The well-known Merkle-Hellman knapsack cryptosystem was first described by Merkle and Hellman in 1978. This public key cryptosystem, and several variants of it, were broken in the early 1980s. In this section we will show how basis reduction can be used to break the Merkle-Hellman knapsack cryptosystem.

The term "knapsack" is actually a misnomer. The Knapsack problem, as it is usually defined, is a problem involving selecting objects with given weights and profits in such a way that a specified capacity is not exceeded and a maximum profit is attained (see Problem 1.4). The Merkle-Hellman knapsack cryptosystem is instead based on Problem 8.2.

Problem 8.2:	Subset Sum
Instance:	positive integers $a_1, \ldots a_n$ and z. The a_is are called *sizes* and z is called the *target sum*.
Find:	a 0-1 vector $U = [u_1, \ldots, u_n]$ such that $$\sum_{i=1}^{n} u_i a_i = z.$$

Problem 8.2 is a search problem which is known to be NP-hard. Among other things, this means that there is no known polynomial-time algorithm that solves it. But even if a problem has no polynomial-time algorithm to solve it in general, this does not rule out the possibility that certain special cases can be solved in polynomial time. This is indeed the situation with the Subset Sum problem.

we define a list of sizes, $[a_1, \ldots, a_n]$, to be *superincreasing* if

$$a_j > \sum_{i=1}^{j-1} a_i$$

for $2 \leq j \leq n$. If the list of sizes is superincreasing, then the Subset Sum problem can be solved very easily in time $O(n)$ by a greedy algorithm, and a solution U (if it exists) must be unique. The algorithm to do this is presented in Algorithm 8.5.

Algorithm 8.5: SUPERINCREASINGSOLVER $(a_1, a_2, \ldots, a_n, z)$

for $i \leftarrow n$ **downto** 1

do { **if** $z \geq a_i$
  then { $z \leftarrow z - a_i$
    $u_i \leftarrow 1$
  else $u_i \leftarrow 0$

if $z = 0$
  then $U \leftarrow [u_1, \ldots, u_n]$ is the solution
  else there is no solution.

Suppose $A = [a_1, \ldots, a_n]$ is superincreasing, and consider the function

$$e_A : \{0,1\}^n \rightarrow \left\{0, \ldots, \sum_{i=1}^{n} a_i\right\}$$

defined by the rule

$$e_A(u_1, \ldots, u_n) = \sum_{i=1}^{n} u_i a_i.$$

Is e_A a possible candidate for an encryption rule? Since A is superincreasing, e_A is an injection, and the algorithm presented in Algorithm 8.5 would be the corresponding decryption algorithm. However, such a system would be completely insecure because anyone can decrypt a message that is encrypted in this way.

The strategy therefore is to transform the list of sizes in such a way that it is no longer superincreasing. The receiver will be able to apply an inverse transformation to restore the superincreasing list of sizes. On the other hand, an observer who does not know the transformation that was applied is faced with what looks like a general, apparently difficult, instance of the Subset Sum problem when he tries to decrypt a ciphertext.

One suitable type of transformation is a *modular transformation*. That is, a prime modulus p is chosen such that

$$p > \sum_{i=1}^{n} a_i,$$

as well as a multiplier m, where $1 \leq m \leq p - 1$. Then we define

$$t_i = ma_i \bmod p,$$

$1 \leq i \leq n$. The list of sizes $T = [t_1, \ldots, t_n]$ will be the public key used for encryption. The values m, p used to define the modular transformation are secret. The complete description of the Merkle-Hellman knapsack cryptosystem is given in Definition 8.3.

Definition 8.3: *Merkle-Hellman knapsack cryptosystem.* Let $A = [a_1, \ldots, a_n]$ be a superincreasing list of integers, let $p > \sum_{i=1}^{n} a_i$ be prime, and let $1 \leq m \leq p-1$. For $1 \leq i \leq n$, define

$$t_i = ma_i \bmod p,$$

and denote $T = [t_1, \ldots, t_n]$. Let the set of plaintext messages be $\mathcal{P} = \{0,1\}^n$, let the set of ciphertext messages be $\mathcal{C} = \{0, \ldots, n(p-1)\}$, and let the set of keys be

$$\mathcal{K} = \{(A, p, m, T)\},$$

where A, p, m, and T are constructed as described above. T is public, and p, m and A are secret.

For $K = (A, p, m, T)$, and $[u_1, \ldots, u_n] \in \mathcal{P}$, define

$$e_K(u_1, \ldots, u_n) = \sum_{i=1}^{n} u_i t_i \in \mathcal{C}.$$

For $y \in \mathcal{C}$, define $z = m^{-1}y \bmod p$ and solve Problem 8.2 with sizes $a_1, \ldots, a_n$ and target sum z, obtaining $d_K(y) = [u_1, \ldots, u_n] \in \mathcal{P}$.

The following small example illustrates the encryption and decryption operations in the Merkle-Hellman knapsack cryptosystem.

Example 8.6

Suppose

$$A = [2, 5, 9, 21, 45, 103, 215, 450, 946]$$

is the secret superincreasing list of sizes. Suppose $p = 2003$ and $m = 1289$. Then the public list of sizes is

$$T = [575, 436, 1586, 1030, 1921, 569, 721, 1183, 1570].$$

Now, if the sender wants to encrypt the plaintext $U = [1, 0, 1, 1, 0, 0, 1, 1, 1]$, she computes

$$y = 575 + 1586 + 1030 + 721 + 1183 + 1570 = 6665.$$

When the ciphertext y is received, we first compute

$$\begin{aligned} z &= a^{-1}y \bmod p \\ &= 317 \times 6665 \bmod 2003 \\ &= 1643. \end{aligned}$$

Then we solve the instance of Problem 8.2, with sizes $s_1, s_2, \ldots, s_n$ and target sum z, using Algorithm 8.5. The plaintext $(1,0,1,1,0,0,1,1,1)$ is obtained. □

By the early 1980s, the Merkle-Hellman knapsack cryptosystem had been broken by Shamir. Shamir was able to use an integer programming algorithm of Lenstra to break the system. This allowed the receivers's trapdoor (or an equivalent trapdoor) to be discovered by a cryptanalyst. This cryptanalyst can decrypt messages exactly as the intended receiver does. To circumvent these attacks on the Merkle-Hellman knapsack cryptosystem other variants were introduced. These variants had the effect of increasing the density of the Subset Sum problem. The *density* of an instance of Problem 8.2, with sizes $A = [a_1, \ldots, a_n]$, is defined by

$$\partial(A) = \frac{n}{\log_2(\max_j a_j)}.$$

If $\partial(A) > 1$, then there will be in general many subsets of the a_is with the same sum. These instances of Problem 8.2 could not be used in a cryptosystem. Consequently the interesting case is when $\partial(A) \leq 1$. Basis reduction can be used to solve almost all instances of Problem 8.2 with $\partial(A)$ sufficiently small.

We begin by reducing Problem 8.2 to Problem 8.1. The method and reasoning is exactly the same as in Section 8.4. However this time the incidence matrix $[a_1, a_2, \ldots, a_n]$ has only one row. The basis is thus

$$M = \left[\begin{array}{cccc|c} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ & & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ \hline a_1 & a_2 & \cdots & a_n & -z \end{array}\right]. \tag{8.7}$$

Example 8.7

In Example 8.6, the public list of sizes is

$$T = [575, 436, 1586, 1030, 1921, 569, 721, 1183, 1570]$$

and the received ciphertext is $y = 6665$. Thus the lattice we wish to reduce is

$$M = \left[\begin{array}{cc} I & \vec{0} \\ T & -y \end{array}\right].$$

Applying Algorithm 8.2, we obtain the basis

$$M' = \left[\begin{array}{rrrrrrrrr|r} -2 & 0 & 1 & 0 & 1 & 0 & 1 & -1 & -2 & 1 \\ -1 & 0 & -1 & 1 & -1 & 0 & 0 & -1 & 0 & 1 \\ 1 & 0 & 0 & -1 & -1 & -1 & 1 & 0 & -1 & 0 \\ 0 & -1 & -2 & -1 & 1 & 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & -1 & -1 & -2 \\ 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 2 & -1 & -1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & -1 & 0 & -1 & 1 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 2 \\ \hline 0 & 1 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 0 \end{array}\right],$$

which has the solution

$$U = [1, 0, 1, 1, 0, 0, 1, 1, 1]$$

in column 7. In fact U is the original plaintext. □

It is interesting that the method of basis reduction breaks the Merkle-Hellman knapsack cryptosystem without determining the multiplier or modulus used.

If the reduced basis obtained by Algorithm 8.2 on the input basis given in Equation 8.7 fails to contain a solution, then further reduction methods such as Algorithm 8.4 can be applied. Alternatively, we can change the form of the basis. A different basis that has been studied is

$$M = \left[\begin{array}{cccc|c} 1 & 0 & \cdots & 0 & \frac{1}{2} \\ 0 & 1 & \cdots & 0 & \frac{1}{2} \\ & & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & \frac{1}{2} \\ \hline a_1 N & a_2 N & \cdots & a_n N & -zN \end{array}\right], \tag{8.8}$$

where $N = \lceil \frac{1}{2}\sqrt{n} \rceil$. Consider the lattice $\mathcal{L}$ with the basis $M = [\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ given in Equation 8.8. If $U = [u_1, u_2, \ldots, u_n]$ is a solution to the Subset Sum problem

$$\sum_{i=1}^{n} u_i a_i = z,$$

then

$$\vec{y} = [\vec{y}_1, \ldots, \vec{y}_n, 0] = \sum_{i=1}^{n} u_i \vec{b}_i - \vec{b}_{n+1}$$

is in $\mathcal{L}$, and $y_i \in \{-\frac{1}{2}, \frac{1}{2}\}$, for $i = 1, 2, \ldots, n$. To recover the solution U from $\vec{y}$ we simply set $u_i = y_i + \frac{1}{2}$ for $i = 1, 2, \ldots, n$. Observe that $||\vec{y}|| = \frac{1}{2}\sqrt{n}$, and so $\vec{y}$ is a vector of short length in $\mathcal{L}(M)$.

TABLE 8.1
Subset Sum data.

	Basis 8.7		Basis 8.8	
∂	LLL	KR	LLL	KR
0.650	55	64	99	100
0.700	38	53	98	98
0.800	24	31	90	87
0.900	19	15	82	79
0.930	13	20	76	83
0.960	12	16	73	78
0.990	10	17	68	78

If the density of the Subset Sum problem is small, then the size of the a_is are large and consequently most of the vectors in the lattice (with basis (8.7) or (8.8)) will have relatively large length. Therefore, an algorithm that reduces the weight of the basis will have a good possibility of finding a solution. In this chapter we have presented two such algorithms, Algorithm 8.2 and Algorithm 8.4. The success in finding the shortest vector in the lattice with either of these algorithms is not guaranteed.

A *lattice oracle* is an algorithm that is guaranteed to return the shortest vector in the lattice in polynomial time. No such algorithm is known. If such an oracle exists, then it has been shown that the shortest vector in the lattice with basis 8.7 corresponds with high probability to the solution of the Subset Sum problem, whenever the density is less than 0.6463. If we instead use basis 8.8, then the shortest vector corresponds to a solution with high probability when the density is less than 0.9408. Therefore, theoretically, basis (8.8) is superior to basis (8.7). To support this, we give experimental evidence in Table 8.1. For each density listed in Table 8.1 we generated 100 random Subset Sum problems of size 20. These were constructed by choosing 20 random non-negative integers less than $\lfloor 2^{20/\partial} \rfloor$. The target sum was created by choosing a random subset of these 20 integers and then summing the entries. In the table we report for each basis and each reduction algorithm the number of successes in solving the 100 random Subset Sum problems.

8.6 Notes

Section 8.2

Analysis similar to that given in this section can be found in [79, 64, 96, 19].

Section 8.3

Algorithm 8.2 appears in [64] where it was introduced as a method for factoring polynomials with rational coefficients. It is often also called the L^3, or the Lovasz algorithm, and it is a crucial component in many number theoretic algorithms [19].

Section 8.4

The multi-row situation when A is the orbit incidence matrix for constructing designs (see Section 6.6.1) was first investigated by Kreher and Radziszowski [58, 60]. In particular Algorithms 8.3 and 8.4 were first described in [58, 59, 60]. To the best of our knowledge no theoretical or experimental analysis has been obtained in this situation. On the other hand, several thousand new combinatorial designs were discovered using the basis reduction algorithm of Kreher and Radziszowski.

Section 8.5

The Merkle-Hellman knapsack cryptosystem was presented in [74]. This system was broken by Shamir [98], and the "iterated" version of the system was broken by Brickell [11]. The analysis using a lattice oracle and giving the bounds on the density appears in the article [23]. For more information, see the survey article by Brickell and Odlyzko [12] and the *Handbook of Applied Cryptography* [73]. Algorithm 8.3 which first appeared in [58, 59] was also used by Schnorr and Euchner to solve Subset Sum problems in [95].

Exercises

8.1 Give a complete proof of Lemma 8.1.

8.2 Which of the following vectors are in the lattice given in Example 8.1?

(a) $[-1, 18]$
(b) $[4, 12]$
(c) $[1, 6]$
(d) $[1, 10]$
(e) $[1, -11]$

8.3 Consider the lattice with basis

$$M = \begin{bmatrix} 1 & -2 \\ 3 & 1 \end{bmatrix}$$

that is displayed in Example 8.1.

(a) Compute $\mathrm{wt}(M)$ and $\mathrm{vol}(\mathcal{L})$ and verify Hadamard's inequality for this lattice.

(b) Show geometrically (i.e., draw a picture) that for this lattice

$$\mathrm{wt}(M) \geq \mathrm{vol}(\mathcal{L}).$$

8.4 Let $[\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n]$ be a basis. Give a formal proof that an operation of the form

$$\text{replace } \vec{b}_j \text{ with } \alpha_1\vec{b}_1 + \alpha_2\vec{b}_2 + \cdots + \vec{b}_j + \cdots + \alpha_n\vec{b}_n$$

can be obtained by performing a sequence of the following three operations:

(a) Replace $\vec{b}_i$ with $\vec{b}_i + \vec{b}_j$,

(b) Replace $\vec{b}_i$ with $-\vec{b}_i + \vec{b}_j$, and

(c) Replace $\vec{b}_i$ with $\vec{b}_i - \vec{b}_j$, where $i \neq j$.

8.5 Using Algorithm 8.1, work out by hand an orthogonal basis for the lattice spanned by

$$M = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

Check your results by computer. What is the volume of the lattice spanned by the columns of M? Verify Lemma 8.4 for the matrix M.

8.6 Consider the matrix

$$M = \begin{bmatrix} 0 & 2 & 3 & 1 \\ 1 & 0 & -1 & 5 \\ 2 & -2 & 2 & 0 \\ 2 & 2 & 0 & -2 \end{bmatrix}.$$

(a) Show that M is a reduced basis.

(b) Verify the inequalities of Theorem 8.8 for the matrix M.

8.7 Use the algorithms in Section 8.4 to construct a Steiner triple system of order 9 that has

$$g = (0, 1, 2)(3, 4, 5)(6, 7, 8)$$

as an automorphism.

8.8 Suppose the Merkle-Hellman knapsack cryptosystem has as its public list of sizes the vector

$$T = [1394, 1256, 1508, 1987, 439, 650, 724, 339, 2303, 810].$$

Suppose an observer discovers that $p = 2503$.

(a) By trial and error, determine the value a such that the list $a^{-1}T \bmod p$ is a permutation of a superincreasing list.

(b) Show how the ciphertext 5746 would be decrypted.

(c) Use basis reduction to decrypt the ciphertext 5746.

8.9 Develop an algorithm similar to Algorithm 8.3 that reduces the weight of the basis by considering combinations of the form

$$\vec{v} = \epsilon_1 \vec{b}_{i_1} + \epsilon_2 \vec{b}_{i_2} + \epsilon_3 \vec{b}_{i_3},$$

where $\epsilon_i = \pm 1$, for $i = 1, 2, 3$.

8.10 For each of the following Subset Sum problems compute their density and use basis reduction to find a solution.

(a)

$$A = [283615655564068, 796478694573302, 600340146256703,$$
$$732983327534134, 786266787523357, 105515816928335,$$
$$1128976272031 31, 330057122934813, 1089988300272331,$$
$$1051338601577848, 1109763392717310, 145117009247205,$$

283635625683684, 6217169571139, 909231046365184,
740552083084632, 767717555811633, 222691570662389,
287870530458475, 250604219988445],

$z = 5055299030829558$

(b)

$A =$ [7960137240, 7503674315, 8975593017, 6982240834, 750319933,
2263778309, 5779454351, 2189761281, 6377653436, 1899000113,
560590007, 6148611908, 5254132888, 4377585063, 1837007135,
8439676091, 4254195333, 5970662702, 1507562435, 1826255982],

$z = 37987557118$

(c)

$A =$ [806109, 408997, 1169428, 1011478, 1150062, 1182254, 658173,
1198146, 1199680, 430790, 774558, 850850, 916096, 1085626,
164865, 288661, 260406, 619265, 1030628, 946958],

$z = 8440889$