Example 7.33. Let $(m_1, \ldots, m_n, S)$ be a knapsack problem. The associated lattice $L_{\boldsymbol{M},S}$ is generated by the rows of the matrix (7.4) given on page 383. The matrix $L_{\boldsymbol{M},S}$ has dimension $n+1$ and determinant $\det L_{\boldsymbol{M},S} = 2^n S$. As explained in Sect. 7.2, the number S satisfies $S = \mathcal{O}(2^{2n})$, so $S^{1/n} \approx 4$. This allows us to approximate the Gaussian shortest length as

$$\begin{aligned}\sigma(L_{\boldsymbol{M},S}) &= \sqrt{\frac{n+1}{2\pi e}}(\det L_{\boldsymbol{M},S})^{1/(n+1)} = \sqrt{\frac{n+1}{2\pi e}}(2^n S)^{1/(n+1)} \\ &\approx \sqrt{\frac{n}{2\pi e}} \cdot 2S^{1/n} \approx \sqrt{\frac{n}{2\pi e}} \cdot 8 \approx 1.936\sqrt{n}.\end{aligned}$$

On the other hand, as explained in Sect. 7.2, the lattice $L_{\boldsymbol{M},S}$ contains a vector $\boldsymbol{t}$ of length $\sqrt{n}$, and knowledge of $\boldsymbol{t}$ reveals the solution to the subset-sum problem. Hence solving SVP for the lattice $L_{\boldsymbol{M},S}$ is very likely to solve the subset-sum problem. For a further discussion of the use of lattice methods to solve subset-sum problems, see Sect. 7.14.2.

We will find that the Gaussian heuristic is useful in quantifying the difficulty of locating short vectors in lattices. In particular, if the actual shortest vector of a particular lattice L is significantly shorter than $\sigma(L)$, then lattice reduction algorithms such as LLL seem to have a much easier time locating the shortest vector.

A similar argument leads to a Gaussian heuristic for CVP. Thus if $L \subset \mathbb{R}^n$ is a random lattice of dimension n and $\boldsymbol{w} \in \mathbb{R}^n$ is a random point, then we expect that the lattice vector $\boldsymbol{v} \in L$ closest to $\boldsymbol{w}$ satisfies

$$\|\boldsymbol{v} - \boldsymbol{w}\| \approx \sigma(L).$$

And just as for SVP, if L contains a point that is significantly closer than $\sigma(L)$ to $\boldsymbol{w}$, then lattice reduction algorithms have an easier time solving CVP.

7.6 Babai's Algorithm and Using a "Good" Basis to Solve apprCVP

If a lattice $L \subset \mathbb{R}^n$ has a basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ consisting of vectors that are pairwise orthogonal, i.e., such that

$$\boldsymbol{v}_i \cdot \boldsymbol{v}_j = 0 \quad \text{for all } i \neq j,$$

then it is easy to solve both SVP and CVP. Thus to solve SVP, we observe that the length of any vector in L is given by the formula

$$\|a_1\boldsymbol{v}_1 + a_2\boldsymbol{v}_2 + \cdots + a_n\boldsymbol{v}_n\|^2 = a_1^2\|\boldsymbol{v}_1\|^2 + a_2^2\|\boldsymbol{v}_2\|^2 + \cdots + a_n^2\|\boldsymbol{v}_n\|^2.$$

Since $a_1, \ldots, a_n \in \mathbb{Z}$, we see that the shortest nonzero vector(s) in L are simply the shortest vector(s) in the set $\{\pm\boldsymbol{v}_1, \ldots, \pm\boldsymbol{v}_n\}$.

Similarly, suppose that we want to find the vector in L that is closest to a given vector $\boldsymbol{w} \in \mathbb{R}^n$. We first write

$$\boldsymbol{w} = t_1\boldsymbol{v}_1 + t_2\boldsymbol{v}_2 + \cdots + t_n\boldsymbol{v}_n \quad \text{with } t_1, \ldots, t_n \in \mathbb{R}.$$

Then for $\boldsymbol{v} = a_1\boldsymbol{v}_1 + \cdots + a_n\boldsymbol{v}_n \in L$, we have

$$\|\boldsymbol{v} - \boldsymbol{w}\|^2 = (a_1 - t_1)^2\|\boldsymbol{v}_1\|^2 + (a_2 - t_2)^2\|\boldsymbol{v}_2\|^2 + \cdots + (a_n - t_n)^2\|\boldsymbol{v}_n\|^2. \quad (7.23)$$

The a_i are required to be integers, so (7.23) is minimized if we take each a_i to be the integer closest to the corresponding t_i.

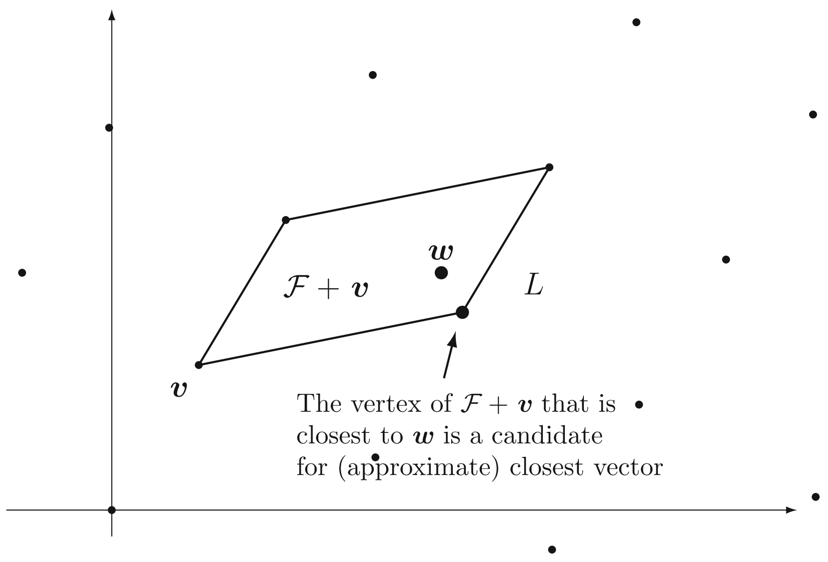
$\boldsymbol{w}$
$\mathcal{F} + \boldsymbol{v}$
L
$\boldsymbol{v}$
The vertex of $\mathcal{F} + \boldsymbol{v}$ that is closest to $\boldsymbol{w}$ is a candidate for (approximate) closest vector

Figure 7.3: Using a given fundamental domain to try to solve CVP

It is tempting to try a similar procedure with an arbitrary basis of L. If the vectors in the basis are reasonably orthogonal to one another, then we are likely to be successful in solving CVP; but if the basis vectors are highly non-orthogonal, then the algorithm does not work well. We briefly discuss the underlying geometry, then describe the general method, and conclude with a 2-dimensional example.

A basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ for L determines a fundamental domain $\mathcal{F}$ in the usual way, see (7.9). Proposition 7.18 says that the translates of $\mathcal{F}$ by the elements of L fill up the entire space $\mathbb{R}^n$, so any $\boldsymbol{w} \in \mathbb{R}^n$ is in a unique translate $\mathcal{F} + \boldsymbol{v}$ of $\mathcal{F}$ by an element $\boldsymbol{v} \in L$. We take the vertex of the parallelepiped $\mathcal{F} + \boldsymbol{v}$ that is closest to $\boldsymbol{w}$ as our hypothetical solution to CVP. This procedure is illustrated in Fig. 7.3. It is easy to find the closest vertex, since

$$\boldsymbol{w} = \boldsymbol{v} + \epsilon_1\boldsymbol{v}_1 + \epsilon_2\boldsymbol{v}_2 + \cdots + \epsilon_n\boldsymbol{v}_n \quad \text{for some } 0 \le \epsilon_1, \epsilon_2, \ldots, \epsilon_n < 1,$$

so we simply replace ϵ_i by 0 if it is less than $\frac{1}{2}$ and replace it by 1 if it is greater than or equal to $\frac{1}{2}$.

Looking at Fig. 7.3 makes it seem that this procedure is bound to work, but that's because the basis vectors in the picture are reasonably orthogonal to one another. Figure 7.4 illustrates two different bases for the same lattice. The first basis is "good" in the sense that the vectors are fairly orthogonal; the second basis is "bad" because the angle between the basis vectors is small.

If we try to solve CVP using a bad basis, we are likely run into problems as illustrated in Fig. 7.5. The nonlattice target point is actually quite close to a lattice point, but the parallelogram is so elongated that the closest vertex

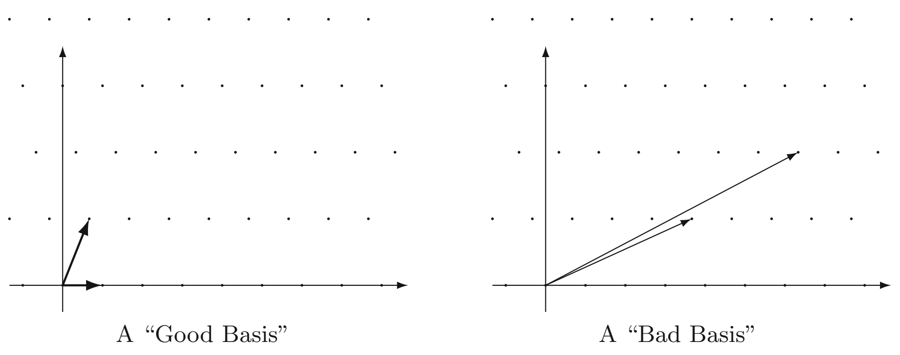

Figure 7.4: Two different bases for the same lattice

to the target point is quite far away. And it is important to note that the difficulties get much worse as the dimension of the lattice increases. Examples visualized in dimension 2 or 3, or even dimension 4 or 5, do not convey the extent to which the following closest vertex algorithm generally fails to solve even apprCVP unless the basis is quite orthogonal.

Theorem 7.34 (Babai's Closest Vertex Algorithm)**.** *Let $L \subset \mathbb{R}^n$ be a lattice with basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$, and let $\boldsymbol{w} \in \mathbb{R}^n$ be an arbitrary vector. If the vectors in the basis are sufficiently orthogonal to one another, then the following algorithm solves* CVP*.*

```
Write $\boldsymbol{w} = t_1\boldsymbol{v}_1 + t_2\boldsymbol{v}_2 + \cdots + t_n\boldsymbol{v}_n$ with $t_1, \ldots, t_n \in \mathbb{R}$.
Set $a_i = \lfloor t_i \rceil$ for $i = 1, 2, \ldots, n$.
Return the vector $\boldsymbol{v} = a_1\boldsymbol{v}_1 + a_2\boldsymbol{v}_2 + \cdots + a_n\boldsymbol{v}_n$.
```

In general, if the vectors in the basis are reasonably orthogonal to one another, then the algorithm solves some version of apprCVP*, but if the basis vectors are highly nonorthogonal, then the vector returned by the algorithm is generally far from the lattice vector that is closest to $\boldsymbol{w}$.*

Example 7.35. Let $L \subset \mathbb{R}^2$ be the lattice given by the basis

$$\boldsymbol{v}_1 = (137, 312) \quad \text{and} \quad \boldsymbol{v}_2 = (215, -187).$$

We are going to use Babai's algorithm (Theorem 7.34) to find a vector in L that is close to the vector

$$\boldsymbol{w} = (53172, 81743).$$

The first step is to express $\boldsymbol{w}$ as a linear combination of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ using real coordinates. We do this using linear algebra. Thus we need to find $t_1, t_2 \in \mathbb{R}$ such that

$$\boldsymbol{w} = t_1\boldsymbol{v}_1 + t_2\boldsymbol{v}_2.$$

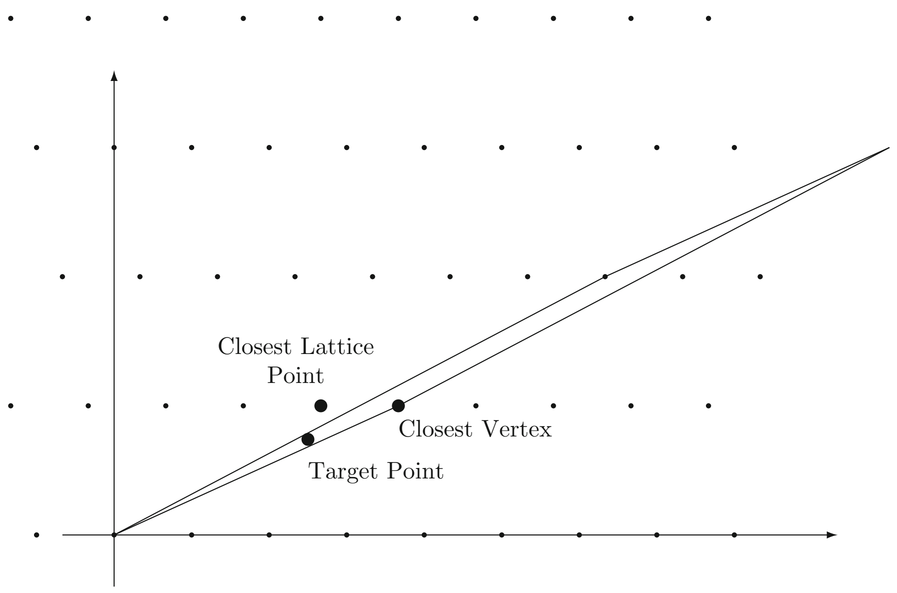

Figure 7.5: Babai's algorithm works poorly if the basis is "bad"

This gives the two linear equations

$$53172 = 137t_1 + 215t_2 \quad \text{and} \quad 81743 = 312t_1 - 187t_2, \tag{7.24}$$

or, for those who prefer matrix notation,

$$(53172, 81743) = (t_1, t_2)\begin{pmatrix} 137 & 312 \\ 215 & -187 \end{pmatrix}. \tag{7.25}$$

It is easy to solve for (t_1, t_2), either by solving the system (7.24) or by inverting the matrix in (7.25). We find that $t_1 \approx 296.85$ and $t_2 \approx 58.15$. Babai's algorithm tells us to round t_1 and t_2 to the nearest integer and then compute

$$\boldsymbol{v} = \lfloor \boldsymbol{t}_1 \rceil \boldsymbol{v}_1 + \lfloor \boldsymbol{t}_2 \rceil \boldsymbol{v}_2 = 297(137, 312) + 58(215, -187) = (53159, 81818).$$

Then $\boldsymbol{v}$ is in L and $\boldsymbol{v}$ should be close to $\boldsymbol{w}$. We find that

$$\|\boldsymbol{v}-\boldsymbol{w}\| \approx 76.12$$

is indeed quite small. This is to be expected, since the vectors in the given basis are fairly orthogonal to one another, as is seen by the fact that the Hadamard ratio

$$\mathcal{H}(\boldsymbol{v}_1,\boldsymbol{v}_2) = \left(\frac{\det(L)}{\|\boldsymbol{v}_1\|\|\boldsymbol{v}_2\|}\right)^{1/2} \approx \left(\frac{92699}{(340.75)(284.95)}\right)^{1/2} \approx 0.977$$

is reasonably close to 1.

We now try to solve the same closest vector problem in the same lattice, but using the new basis

$$\boldsymbol{v}_1' = (1975, 438) = 5\boldsymbol{v}_1 + 6\boldsymbol{v}_2 \quad \text{and} \quad \boldsymbol{v}_2' = (7548, 1627) = 19\boldsymbol{v}_1 + 23\boldsymbol{v}_2.$$

The system of linear equations

$$(53172, 81743) = (t_1, t_2)\begin{pmatrix} 1975 & 438 \\ 7548 & 1627 \end{pmatrix} \tag{7.26}$$

has the solution $(t_1,t_2) \approx (5722.66, -1490.34)$, so we set

$$\boldsymbol{v}' = 5723\boldsymbol{v}_1' - 1490\boldsymbol{v}_2' = (56405, 82444).$$

Then $\boldsymbol{v}' \in L$, but $\boldsymbol{v}'$ is not particularly close to $\boldsymbol{w}$, since

$$\|\boldsymbol{v}'-\boldsymbol{w}\| \approx 3308.12.$$

The nonorthogonality of the basis $\{\boldsymbol{v}_1', \boldsymbol{v}_2'\}$ is shown by the smallness of the Hadamard ratio

$$\mathcal{H}(\boldsymbol{v}_1',\boldsymbol{v}_2') = \left(\frac{\det(L)}{\|\boldsymbol{v}_1\|\|\boldsymbol{v}_2\|}\right)^{1/2} \approx \left(\frac{92699}{(2022.99)(7721.36)}\right)^{1/2} \approx 0.077.$$

7.7 Cryptosystems Based on Hard Lattice Problems

During the mid-1990s, several cryptosystems were introduced whose underlying hard problem was SVP and/or CVP in a lattice L of large dimension n. The most important of these, in alphabetical order, were the Ajtai–Dwork cryptosystem [4], the GGH cryptosystem of Goldreich, Goldwasser, and Halevi [49], and the NTRU cryptosystem proposed by Hoffstein, Pipher, and Silverman [54].

The motivation for the introduction of these cryptosystems was twofold. First, it is certainly of interest to have cryptosystems based on a variety of hard mathematical problems, since then a breakthrough in solving one mathematical problem does not compromise the security of all systems. Second,