NTRU is easily transformed into an SVP (for key recovery) or a CVP (for plaintext recovery) in a special class of lattices. The NTRU lattices, which are described in Sect. 7.11, are lattices of even dimension $n = 2N$ consisting of all vectors $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z}^{2N}$ satisfying

$$\boldsymbol{y} \equiv \boldsymbol{x}H \pmod{q}$$

for some fixed positive integer q that is a public parameter. (In practice, $q = \mathcal{O}(n)$.) The matrix H, which is the public key, is an N-by-N circulant matrix. This means that each successive row of H is a rotation of the previous row, so in order to describe H, it suffices to specify its first row. Thus the public key has size $\mathcal{O}(n \log n)$, which is significantly smaller than GGH.

The NTRU private key is a single short vector $(\boldsymbol{f}, \boldsymbol{g}) \in L$. The set consisting of the short vector $(\boldsymbol{f}, \boldsymbol{g})$, together with its partial rotations, gives $N = \frac{1}{2}\dim(L)$ independent short vectors in L. This allows the owner of $(\boldsymbol{f}, \boldsymbol{g})$ to solve certain instances of CVP in L and thereby recover the encrypted plaintext. (For details, see Sect. 7.11 and Exercise 7.36.) Thus the security of the plaintext relies on the difficulty of solving CVP in the NTRU lattice. Further, the vector $(\boldsymbol{f}, \boldsymbol{g})$ and its rotations are almost certainly the shortest nonzero vectors in L, so NTRU is also vulnerable to a solution of SVP.

7.8 The GGH Public Key Cryptosystem

Alice begins by choosing a set of linearly independent vectors

$$\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n \in \mathbb{Z}^n$$

that are reasonably orthogonal to one another. One way to do this is to fix a parameter d and choose the coordinates of $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ randomly between $-d$ and d. Alice can check that her choice of vectors is good by computing the Hadamard ratio (Remark 7.27) of her basis and verifying that it is not too small. The vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ are Alice's private key. For convenience, we let V be the n-by-n matrix whose rows are the vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$, and we let L be the lattice generated by these vectors.

Alice next chooses an n-by-n matrix U with integer coefficients and $\det(U) = \pm 1$. One way to create U is as a product of a large number of randomly chosen elementary matrices. She then computes

$$W = UV.$$

The row vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ of W are a new basis for L. They are Alice's public key.

When Bob wants to send a message to Alice, he selects a small vector $\boldsymbol{m}$ with integer coordinates as his plaintext, e.g., $\boldsymbol{m}$ might be a binary vector. Bob also chooses a small random perturbation vector $\boldsymbol{r}$ that acts as a random element. For example, he might choose the coordinates of $\boldsymbol{r}$ randomly

between $-\delta$ and δ, where δ is a fixed public parameter. He then computes the vector

$$\boldsymbol{e} = \boldsymbol{m}W + \boldsymbol{r} = \sum_{i=1}^{n} m_i \boldsymbol{w}_i + \boldsymbol{r},$$

which is his ciphertext. Notice that $\boldsymbol{e}$ is not a lattice point, but it is close to the lattice point $\boldsymbol{m}W$, since $\boldsymbol{r}$ is small.

Alice	**Bob**
Key creation	
Choose a good basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$. Choose an integer matrix U satisfying $\det(U) = \pm 1$. Compute a bad basis $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$ as the rows of $W = UV$. Publish the public key $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n$.	
Encryption	
	Choose small plaintext vector $\boldsymbol{m}$. Choose random small vector $\boldsymbol{r}$. Use Alice's public key to compute $\boldsymbol{e} = x_1\boldsymbol{w}_1 + \cdots + x_n\boldsymbol{w}_n + \boldsymbol{r}$. Send the ciphertext $\boldsymbol{e}$ to Alice.
Decryption	
Use Babai's algorithm to compute the vector $\boldsymbol{v} \in L$ closest to $\boldsymbol{e}$. Compute $\boldsymbol{v}W^{-1}$ to recover $\boldsymbol{m}$.	

Table 7.3: The GGH cryptosystem

Decryption is straightforward. Alice uses Babai's algorithm, as described in Theorem 7.34, with the good basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ to find a vector in L that is close to $\boldsymbol{e}$. Since she is using a good basis and $\boldsymbol{r}$ is small, the lattice vector that she finds is $\boldsymbol{m}W$. She then multiplies by W^{-1} to recover $\boldsymbol{m}$. The GGH cryptosystem is summarized in Table 7.3.

Example 7.36. We illustrate the GGH cryptosystem with a 3-dimensional example. For Alice's private good basis we take

$$\boldsymbol{v}_1 = (-97, 19, 19), \quad \boldsymbol{v}_2 = (-36, 30, 86), \quad \boldsymbol{v}_3 = (-184, -64, 78).$$

The lattice L spanned by $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, and $\boldsymbol{v}_3$ has determinant $\det(L) = 859516$, and the Hadamard ratio of the basis is

$$\mathcal{H}(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3) = \big(\det(L)/\|\boldsymbol{v}_1\|\,\|\boldsymbol{v}_2\|\,\|\boldsymbol{v}_3\|\big)^{1/3} \approx 0.74620.$$

Alice multiplies her private basis by the matrix

$$U = \begin{pmatrix} 4327 & -15447 & 23454 \\ 3297 & -11770 & 17871 \\ 5464 & -19506 & 29617 \end{pmatrix},$$

which has determinant $\det(U) = -1$, to create her public basis

$$\begin{aligned} \boldsymbol{w}_1 &= (-4179163, -1882253, 583183), \\ \boldsymbol{w}_2 &= (-3184353, -1434201, 444361), \\ \boldsymbol{w}_3 &= (-5277320, -2376852, 736426). \end{aligned}$$

The Hadamard ratio of the public basis is very small,

$$\mathcal{H}(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3) = \bigl(\det(L)/\|\boldsymbol{w}_1\|\,\|\boldsymbol{w}_2\|\,\|\boldsymbol{w}_3\|\bigr)^{1/3} \approx 0.0000208.$$

Bob decides to send Alice the plaintext $\boldsymbol{m} = (86, -35, -32)$ using the random element $\boldsymbol{r} = (-4, -3, 2)$. The corresponding ciphertext is

$$\begin{aligned} \boldsymbol{e} &= (86, -35, -32) \begin{pmatrix} -4179163 & -1882253 & 583183 \\ -3184353 & -1434201 & 444361 \\ -5277320 & -2376852 & 736426 \end{pmatrix} + (-4, -3, 2) \\ &= (-79081427, -35617462, 11035473). \end{aligned}$$

Alice uses Babai's algorithm to decrypt. She first writes $\boldsymbol{e}$ as a linear combination of her private basis with real coefficients,

$$\boldsymbol{e} \approx 81878.97\boldsymbol{v}_1 - 292300.00\boldsymbol{v}_2 + 443815.04\boldsymbol{v}_3.$$

She rounds the coefficients to the nearest integer and computes a lattice vector

$$\boldsymbol{v} = 81879\boldsymbol{v}_1 - 292300\boldsymbol{v}_2 + 443815\boldsymbol{v}_3 = (-79081423, -35617459, 11035471)$$

that is close to $\boldsymbol{e}$. She then recovers $\boldsymbol{m}$ by expressing $\boldsymbol{v}$ as a linear combination of the public basis and reading off the coefficients,

$$\boldsymbol{v} = 86\boldsymbol{w}_1 - 35\boldsymbol{w}_2 - 32\boldsymbol{w}_3.$$

Now suppose that Eve tries to decrypt Bob's message, but she knows only the public basis $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3$. If she applies Babai's algorithm using the public basis, she finds that

$$\boldsymbol{e} \approx 75.76\boldsymbol{w}_1 - 34.52\boldsymbol{w}_2 - 24.18\boldsymbol{w}_3.$$

Rounding, she obtains a lattice vector

$$\boldsymbol{v}' = 76\boldsymbol{w}_1 - 35\boldsymbol{w}_2 - 24\boldsymbol{w}_3 = (-79508353, -35809745, 11095049)$$

that is somewhat close to $\boldsymbol{e}$. However, this lattice vector gives the incorrect plaintext $(76, -35, -24)$, not the correct plaintext $\boldsymbol{m} = (86, -35, -32)$. It is instructive to compare how well Babai's algorithm did for the different bases. We find that

$$\|\boldsymbol{e} - \boldsymbol{v}\| \approx 5.39 \qquad \text{and} \qquad \|\boldsymbol{e} - \boldsymbol{v}'\| \approx 472004.09$$

Of course, the GGH cryptosystem is not secure in dimension 3, since even if we use numbers that are large enough to make an exhaustive search impractical, there are efficient algorithms to find good bases in low dimension. In dimension 2, an algorithm for finding a good basis dates back to Gauss. A powerful generalization to arbitrary dimension, known as the LLL algorithm, is covered in Sect. 7.13.

Remark 7.37. We observe that GGH is an example of a probabilistic cryptosystem (see Sect. 3.10), since a single plaintext leads to many different ciphertexts due to the choice of the random perturbation $\boldsymbol{r}$. This leads to a potential danger if Bob sends the same message twice using different random perturbations, or sends different messages using the same random perturbation. One possible solution is to choose the random perturbation $\boldsymbol{r}$ deterministically by applying a hash function (Sect. 8.1) to the plaintext $\boldsymbol{m}$, but this causes other security issues. See Exercises 7.20 and 7.21 for a further discussion.

Remark 7.38. An alternative version of GGH reverses the roles of $\boldsymbol{m}$ and $\boldsymbol{r}$, so the ciphertext has the form $\boldsymbol{e} = \boldsymbol{r}W + \boldsymbol{m}$. Alice finds $\boldsymbol{r}W$ by computing the lattice vector closest to $\boldsymbol{e}$, and then she recovers the plaintext as $\boldsymbol{m} = \boldsymbol{e} - \boldsymbol{r}W$.

7.9 Convolution Polynomial Rings

In this section we describe the special sort of polynomial quotient rings that are used by the NTRU public key cryptosystem, which is the topic of Sects. 7.10 and 7.11. The reader who is unfamiliar with basic ring theory should read Sect. 2.10 before continuing.

Definition. Fix a positive integer N. The *ring of convolution polynomials* (*of rank* N) is the quotient ring

$$R = \frac{\mathbb{Z}[x]}{(x^N - 1)}.$$

Similarly, the *ring of convolution polynomials* (*modulo* q) is the quotient ring

$$R_q = \frac{(\mathbb{Z}/q\mathbb{Z})[x]}{(x^N - 1)}.$$

Proposition 2.50 tells us that every element of R or R_q has a unique representative of the form

$$a_0 + a_1 x + a_2 x^2 + \cdots + a_{N-1} x^{N-1}$$

with the coefficients in $\mathbb{Z}$ or $\mathbb{Z}/q\mathbb{Z}$, respectively. We observe that it is easier to do computations in the rings R and R_q than it is in more general polynomial quotient rings, because the polynomial $x^N - 1$ has such a simple form. The