Returning to Example 7.56, we see that the expected number of decryption keys in $\mathcal{T}(84,83)$ for $N = 251$ and $q = 257$ is

$$\left(\frac{3}{257}\right)^{251}\binom{251}{84}\binom{167}{83} \approx 2^{-1222.02}. \tag{7.45}$$

Of course, if $\boldsymbol{h}(x)$ is an NTRUEncrypt public key, then there do exist decryption keys, since we built the decryption key $\boldsymbol{f}(x)$ into the construction of $\boldsymbol{h}(x)$. But the probability calculation (7.45) makes it unlikely that there are any additional decryption keys beyond $\boldsymbol{f}(x)$ and its rotations.

7.11 NTRUEncrypt as a Lattice Cryptosystem

In this section we explain how NTRU key recovery can be formulated as a shortest vector problem in a certain special sort of lattice. Exercise 7.36 sketches a similar description of NTRU plaintext recovery as a closest vector problem.

7.11.1 The NTRU Lattice

Let

$$\boldsymbol{h}(x) = h_0 + h_1x + \cdots + h_{N-1}x^{N-1}$$

be an NTRUEncrypt public key. The *NTRU lattice* $L_{\boldsymbol{h}}^{\text{NTRU}}$ associated to $\boldsymbol{h}(x)$ is the $2N$-dimensional lattice spanned by the rows of the matrix

$$M_{\boldsymbol{h}}^{\text{NTRU}} = \left(\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right).$$

Notice that $M_{\boldsymbol{h}}^{\text{NTRU}}$ is composed of four N-by-N blocks:

$$\begin{aligned} \text{Upper left block} &= \text{Identity matrix}, \\ \text{Lower left block} &= \text{Zero matrix}, \\ \text{Lower right block} &= q \text{ times the identity matrix}, \\ \text{Upper right block} &= \text{Cyclical permutations of the coefficients of } \boldsymbol{h}(x). \end{aligned}$$

It is often convenient to abbreviate the NTRU matrix as

$$M_{\boldsymbol{h}}^{\mathrm{NTRU}} = \begin{pmatrix} I & \boldsymbol{h} \\ 0 & qI \end{pmatrix}, \tag{7.46}$$

where we view (7.46) as a 2-by-2 matrix with coefficients in R.

We are going to identify each pair of polynomials

$$\boldsymbol{a}(x) = a_0 + a_1x + \cdots + a_{N-1}x^{N-1} \quad \text{and} \quad \boldsymbol{b}(x) = b_0 + b_1x + \cdots + b_{N-1}x^{N-1}$$

in R with a $2N$-dimensional vector

$$(\boldsymbol{a}, \boldsymbol{b}) = (a_0, a_1, \ldots, a_{N-1}, b_0, b_1, \ldots, b_{N-1}) \in \mathbb{Z}^{2N}.$$

We now suppose that the NTRUEncrypt public key $\boldsymbol{h}(x)$ was created using the private polynomials $\boldsymbol{f}(x)$ and $\boldsymbol{g}(x)$ and compute what happens when we multiply the NTRU matrix by a carefully chosen vector.

Proposition 7.59. *Assuming that* $\boldsymbol{f}(x) \star \boldsymbol{h}(x) \equiv \boldsymbol{g}(x) \pmod{q}$*, let* $\boldsymbol{u}(x) \in R$ *be the polynomial satisfying*

$$\boldsymbol{f}(x) \star \boldsymbol{h}(x) = \boldsymbol{g}(x) + q\boldsymbol{u}(x). \tag{7.47}$$

Then

$$(\boldsymbol{f}, -\boldsymbol{u}) M_{\boldsymbol{h}}^{\mathrm{NTRU}} = (\boldsymbol{f}, \boldsymbol{g}), \tag{7.48}$$

so the vector $(\boldsymbol{f}, \boldsymbol{g})$ *is in the* NTRU *lattice* $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$.

Proof. It is clear that the first N coordinates of the product (7.48) are the vector $\boldsymbol{f}$, since the left-hand side of $M_{\boldsymbol{h}}^{\mathrm{NTRU}}$ is the identity matrix atop the zero matrix. Next consider what happens when we multiply the column of $M_{\boldsymbol{h}}^{\mathrm{NTRU}}$ whose top entry is h_k by the vector $(\boldsymbol{f}, -\boldsymbol{u})$. We get the quantity

$$h_kf_0 + h_{k-1}f_1 + \cdots + h_{k+1}f_{N-1} - qu_k,$$

which is the kth entry of the vector $\boldsymbol{f}(x) \star \boldsymbol{h}(x) - q\boldsymbol{u}(x)$. From (7.47), this is the kth entry of the vector $\boldsymbol{g}$, so the second N coordinates of the product (7.48) form the vector $\boldsymbol{g}$. Finally, (7.48) says that we can get the vector $(\boldsymbol{f}, \boldsymbol{g})$ by taking a certain linear combination of the rows of $M_{\boldsymbol{h}}^{\mathrm{NTRU}}$. Hence $(\boldsymbol{f}, \boldsymbol{g}) \in L_{\boldsymbol{h}}^{\mathrm{NTRU}}$. □

Remark 7.60. Using the abbreviation (7.46) and multiplying 2-by-2 matrices having coefficients in R, the proof of Proposition 7.59 becomes the succinct computation

$$(\boldsymbol{f}, -\boldsymbol{u}) \begin{pmatrix} 1 & \boldsymbol{h} \\ 0 & q \end{pmatrix} = (\boldsymbol{f}, \boldsymbol{f} \star \boldsymbol{h} - q\boldsymbol{u}) = (\boldsymbol{f}, \boldsymbol{g}).$$

Proposition 7.61. *Let* (N, p, q, d) *be NTRUEncrypt parameters, where for simplicity we will assume that*

$$p = 3 \qquad \textit{and} \qquad d \approx N/3 \qquad \textit{and} \qquad q \approx 6pd \approx 2pN.$$

Let $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$ *be an NTRU lattice associated to the private key* $(\boldsymbol{f}, \boldsymbol{g})$.

(a) $\det(L_{\boldsymbol{h}}^{\mathrm{NTRU}}) = q^N$.
(b) $\|(\boldsymbol{f},\boldsymbol{g})\| \approx \sqrt{4d} \approx \sqrt{4N/3} \approx 1.155\sqrt{N}$.
(c) *The Gaussian heuristic predicts that the shortest nonzero vector in the* NTRU *lattice has length*

$$\sigma\bigl(L_{\boldsymbol{h}}^{\mathrm{NTRU}}\bigr) \approx \sqrt{Nq/\pi e} \approx 0.838N.$$

Hence if N is large, then there is a high probability that the shortest nonzero vectors in $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$ are $(\boldsymbol{f},\boldsymbol{g})$ and its rotations. Further,

$$\frac{\|(\boldsymbol{f},\boldsymbol{g})\|}{\sigma(L)} \approx \frac{1.38}{\sqrt{N}},$$

so the vector $(\boldsymbol{f},\boldsymbol{g})$ is a factor of $\mathcal{O}(1/\sqrt{N})$ shorter than predicted by the Gaussian heuristic.

Proof. (a) Proposition 7.20 says that $\det(L_{\boldsymbol{h}}^{\mathrm{NTRU}})$ is equal to the determinant of the matrix $M_{\boldsymbol{h}}^{\mathrm{NTRU}}$. The matrix is upper triangular, so its determinant is the product of the diagonal entries, which equals q^N.
(b) Each of $\boldsymbol{f}$ and $\boldsymbol{g}$ has (approximately) d coordinates equal to 1 and d coordinates equal to -1.
(c) Using (a) and keeping in mind that $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$ has dimension $2N$, we estimate the Gaussian expected shortest length using the formula (7.21),

$$\sigma\bigl(L_{\boldsymbol{h}}^{\mathrm{NTRU}}\bigr) = \sqrt{\frac{2N}{2\pi e}}(\det L)^{1/2N} = \sqrt{\frac{Nq}{\pi e}} \approx \sqrt{\frac{6}{\pi e}}\,N. \qquad \square$$

7.11.2 Quantifying the Security of an NTRU Lattice

Proposition 7.61 says that Eve can determine Alice's private NTRU key if she can find a shortest vector in the NTRU lattice $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$. Thus the security of NTRUEncrypt depends at least on the difficulty of solving SVP in $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$. More generally, if Eve can solve apprSVP in $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$ to within a factor of approximately N^{ϵ} for some $\epsilon < \frac{1}{2}$, then the short vector that she finds will probably serve as a decryption key.

This leads to the question of how to estimate the difficulty of finding a short, or shortest, vector in an NTRU lattice. The LLL algorithm that we describe in Sect. 7.13.2 runs in polynomial time and solves apprSVP to within a factor of 2^N, but if N is large, LLL does not find very small vectors in $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$. In Sect. 7.13.4 we describe a generalization of the LLL algorithm, called BKZ-LLL, that is able to find very small vectors. The BKZ-LLL algorithm includes a blocksize parameter β, and it solves apprSVP to within a factor of $\beta^{2N/\beta}$, but its running time is exponential in β.

Unfortunately, the operating characteristics of standard lattice reduction algorithms such as BKZ-LLL are not nearly as well understood as are the operating characteristics of sieves, the index calculus, or Pollard's ρ method. This makes it difficult to predict theoretically how well a lattice reduction algorithm will perform on any given class of lattices. Thus in practice, the security of a lattice-based cryptosystem such as NTRUEncrypt must be determined experimentally.

Roughly, one takes a sequence of parameters (N, q, d) in which N grows and such that certain ratios involving N, q, and d are held approximately constant. For each set of parameters, one runs many experiments using BKZ-LLL with increasing block size β until the algorithm finds a short vector in $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$. Then one plots the logarithm of the average running time against N, verifies that the points approximately lie on line, and computes the best-fitting line

$$\log(\text{Running Time}) = AN + B. \tag{7.49}$$

After doing this for many values of N up to the point at which the computations become infeasible, one can use the line (7.49) to extrapolate the expected amount of time it would take to find a private key vector in an NTRU lattice $L_{\boldsymbol{h}}^{\mathrm{NTRU}}$ for larger values of N. Such experiments suggest that values of N in the range from 250 to 1000 yield security levels comparable to currently secure implementations of RSA, Elgamal, and ECC. Details of such experiments are described in [102].

Remark 7.62. Proposition 7.61 says that the short target vectors in an NTRU lattice are $\mathcal{O}(\sqrt{N})$ shorter than predicted by the Gaussian heuristic. Theoretically and experimentally, it is true that if a lattice of dimension n has a vector that is extremely small, say $\mathcal{O}(2^n)$ shorter than the Gaussian prediction, then lattice reduction algorithms such as LLL and its variants are very good at finding the tiny vector. It is a natural and extremely interesting question to ask whether vectors that are only $\mathcal{O}(n^\epsilon)$ shorter than the Gaussian prediction might similarly be easier to find. At this time, no one knows the answer to this question.

7.12 Lattice-Based Digital Signature Schemes

We have already seen digital signatures schemes whose security depends on the integer factorization problem (Sect. 4.2) and on the discrete logarithm problem in the multiplicative group (Sect. 4.3) or in an elliptic curve (Sect. 6.4.3). In this section we briefly discuss how digital signature schemes may be constructed from hard lattice problems.

7.12.1 The GGH Digital Signature Scheme

It is easy to convert the CVP idea underlying GGH encryption into a lattice-based digital signature scheme. Samantha knows a good (i.e., short and