# 5. Quantum Resistant Cryptosystems

*I think I can safely say that nobody understands quantum mechanics.*

Richard Feynman (1918–1988)
The 1965 Nobel Laureate in Physics

In this last chapter of the book, we shall introduce some cryptographic systems that resist all known quantum-computing attacks.

## 5.1 Quantum-Computing Attack Resistant

We have seen from previous three chapters that quantum computers, if can be built, can solve the famous infeasible IFP, DLP, and ECDLP problems efficiently in polynomial time, and more importantly, all the cryptographic systems and protocols, such as RSA, DHM, and ECC, based on these three types of infeasible problems can be broken in polynomial time. This might lead to the following wrong believing that quantum computers would speed up all computations and would solve all infeasible computational problems and break all cryptographic systems and protocols. It must be pointed out that quantum computers are not fast versions of classical computers, but just use a different and nonclassical paradigm for computation. They would speed up the computation for some problems such as IFP, DLP, and ECDLP by a large factor. However, for some other infeasible problems such as the famous traveling salesman problem and the shortest lattice problem, their computation power would just be the same as that of any classical computers. In fact, quantum computers have not been shown to solve any $\mathcal{NP}$-complete problems so far. Thus, the cryptographic systems and protocols based on some other infeasible problems, rather than IFP, DLP, and ECDLP, should be or may still be secure. That is, the quantum-computing based attacks would be invalid and no use for those cryptographic systems and protocols, whose security does not rely on the infeasibility of IFP, DLP, and ECDLP. More specifically, quantum computing is good at finding period, since fast Fourier transform (FFT) can be used to compute the period of a function, which is in turn can be extended to be a quantum Fourier transform (QFT) and can be run on a quantum computer. Thus, basically, quantum computers can speed up the computation for any periodic functions, as soon as FFT can be used. On the other hand, if the function is not periodic, or the computation is not suited for applying FFT, then the computation cannot generally be speed up by a quantum computer. As a consequence, any cryptographic systems and protocols, whose security does not rely on periodic problems or functions where FFT cannot be applied, should be potentially quantum-computing attack resistant.

**Problems for Sect. 5.1**

1. What is the main difference between a classical computer and a quantum computer?
2. In what sense or in which case a quantum computer can run fast than a classical computer?
3. Can you find a $\mathcal{NP}$-complete problem which can be solved by a quantum computer in polynomial time?
4. Explain why quantum computers are more powerful on solving the periodic functions. Justify your answer.
5. Explain why there must be many quantum-computing attack resistant cryptographic systems and protocols?

## 5.2 Coding-Based Cryptosystems

In this section, we introduce the most famous code-based cryptosystem, the McEliece system, invented by McEliece in [40]. One of the most important features of the McEliece system is that it has resisted cryptanalysis to date; it is even quantum computer resisted. The idea of the McEliece system is based on coding theory, and its security is based on the fact that decoding an arbitrary linear code is $\mathcal{NP}$-complete.

**Algorithm 5.1 (McEliece's Code-Based Cryptography).** Suppose Bob wishes to send an encrypted message to Alice using Alice's public-key. Alice generates her public-key and the corresponding private-key. Bob uses her public-key to encrypt his message and sends it to Alice; Alice uses her own private-key to decrypt Bob's message.

[1] **Key Generation:** Alice performs:

[1-1]Choose integers $k, n, t$ as common system parameters.

[1-2]Choose a $k \times n$ generator matrix $G$ for a binary $(n, k)$-linear code which can correct $t$ errors and for which an efficient decoding algorithm exists.

[1-3]Select a random $k \times k$ binary non-singular matrix $S$.

[1-4]Select a random $k \times k$ permutation matrix $P$.

[1-5]Compute the $k \times n$ matrix $\widehat{G} = SGP$.

[1-6]Now $(\widehat{G}, t)$ is Alice's public-key whereas $(S, G, P)$ is Alice's private-key.

[2] **Encryption:** Bob uses Alice's public-key to encrypt his message to Alice. Bob performs:

[2-1]Obtain Alice's authentic public key $(\widehat{G}, t)$.

[2-2]Represent the message in binary string $m$ of length $k$.

[2-3]Choose a random binary error vector $z$ of length $n$ having at most $t$ $1$'s.

[2-4]Compute the binary vector $c = m\widehat{G} + z$.

[2-5]Send the ciphertext $c$ to Alice.

[3] **Decryption:** Alice receives Bob's message $m$ and uses her private-key to recover $c$ from $m$. Alice does performs:

[3-1] Compute $\widehat{c} = cP^{-1}$, where $P^{-1}$ is the inverse of the matrix $P$.

[3-2] Use the decoding algorithm for the code generated by $G$ to decode $\widehat{c}$ to $\widehat{m}$.

[3-3] Compute $m = \widehat{m}\widehat{S}^{-1}$. This $m$ is thus the original plaintext.

**Theorem 5.1 (Correctness of McEliece's Cryptosystem).** In McEliece's cryptosystem, $m$ can be correctly recovered from $c$.

**Proof.** Since

$$\begin{aligned}
\widehat{c} &= cP^{-1} \\
&= (m\widehat{G} + z)P^{-1} \\
&= (mSGP + z)P^{-1} \\
&= (mS)G + zP^{-1}, \quad (zP^{-1} \text{ is a vector with at most } t \ 1's)
\end{aligned}$$

the decoding algorithm for the code generated by $G$ corrects $\widehat{c}$ to $\widehat{m} = mS$. Now applying $S^{-1}$ to $\widehat{m}$, we get $mSS^{-1} = m$, the required original plaintext. □

**Remark 5.1.** The security of McEliece's cryptosystem is based on error-correcting codes, particularly the Goppa [39]; if the Goppa code is replaced by other error-correcting codes, the security will be severely weakened. The McEliece's cryptosystem has two main drawbacks:

(1) The public-key is very large.

(2) There is a message expansion by a factor of $n/k$.

It is suggested that the values for the system parameters should be $n = 1024$, $t = 50$, and $k \geqslant 644$. Thus, for these recommended values of system parameters, the public-key has about $2^{19}$ bits, and the message expansion is about 1.6. For these reasons, McEliece's cryptosystem receives little attention in practice. However, as McEliece's cryptosystem is the first probabilistic encryption and, more importantly, it has resisted all cryptanalysis including quantum cryptanalysis, it may be a good candidate to replace RSA in the post-quantum cryptography age.

### Problems for Sect. 5.2

1. Compare the main parameters (such as encryption and decryption complexity, cryptographic resistance, easy to use, secret-key size, and public-key size) of RSA and McEliece systems.
2. Show that decoding a general algebraic code is $\mathcal{NP}$-complete.
3. Write an essay on all possible attacks for the McEliece coding-based cryptosystem.

## 5.3 Lattice-Based Cryptosystems

Cryptography based on ring properties and particularly lattice reduction is another promising direction for post-quantum cryptography, as lattice reduction is a reasonably well-studied hard problem that is currently not known to be solved in polynomial time or even subexponential time on a quantum computer. There are many types of cryptographic systems based on lattice reduction. In this section, we give a brief account of one of the lattice based on cryptographic systems, the NTRU encryption scheme. NTRU is rumored to stand for Nth-degree TRUncated polynomial ring, or Number Theorists aRe Us. It is a rather young cryptosystem, developed by Hoffstein, Pipher, and Silverman [26] in 1995. We give a brief introduction to NTRU; for more information, it can be found in [26, 27].

**Algorithm 5.2 (NTRU Encryption Scheme).** The NTRU encryption scheme works as follows.

[1] **Key Generation:**

[1-1] Randomly generate polynomials $f$ and $g$ in $D_f$ and $D_g$, respectively, each of the form:

$$a(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{N-2}x^{N-2} + a_{N-1}x^{N-1}.$$

[1-2] Invert $f$ in $\mathcal{R}_p$ to obtain $f_p$, and check that $g$ is invertible in $f_q$.

[1-3] The public-key is $h \equiv p \cdot g \cdot f_q \pmod q$. The private-key is the pair $(f, f_p)$.

[2] **Encryption:**

[2-1] Randomly select a small polynomials $r$ in $D_r$.

[2-2] Compute the ciphertext $c \equiv r \cdot h + m \pmod q$.

[3] **Decryption:**

[3-1] Compute $a = \text{center}(f \cdot c)$,

[3-2] Recover $m$ from $c$ by computing $m \equiv f_p \cdot a \pmod q$. This is true since

$$a \equiv p \cdot r \cdot \equiv + f \cdot m \pmod q.$$

In Table 5.1, we present some information comparing NTRU to RSA and McEliece.

**Table 5.1.** Comparison among NTRU, RSA and McEliece

| | NTRU | RSA | McEliece |
|---|---|---|---|
| Encryption speed | $N^2$ | $N^2 \approx N^3$ | $N^2$ |
| Decryption speed | $N^2$ | $N^3$ | $N^2$ |
| Public-key | $N$ | $N$ | $N^2$ |
| Secret-key | $N$ | $N$ | $N^2$ |
| Message expansion | $\log_p q - 1$ | $1 - 1$ | $1 - 1.6$ |

**Problems for Sect. 5.3**

1. Give a critical analysis of the computational complexity of the NTRU cryptosystem.
2. NTRU is currently considered quantum resistant. Show that NTRU is indeed quantum resistant or may not be quantum resistant.
3. Lattice-based cryptography is considered to be quantum resistant. However, if designed not properly, it may be broken by traditional mathematical attacks without using any quantum techniques. For example, the Cai-Cusick lattice-based cryptosystem [17] was recently cracked completely by Pan and Deng [45]. Show that the Cai-Cusick lattice-based cryptosystem can be broken in Polynomial time by classical mathematical attacks.
4. It is widely considered that the multivariate public-key cryptosystems (MPKC, see [20]) are quantum resistant. As the usual approach to polynomial evaluation is FFT like, whereas quantum computation makes a good use of FFT to speed up the computation. With this regard, show that MPKC may not be quantum resistant.

## 5.4 Quantum Cryptosystems

It is evident that if a practical quantum computer is available, then all public-key cryptographic systems based on the difficulty of IFP, DLP, and ECDLP will be insecure. However, the cryptographic systems based on quantum mechanics will still be secure even if a quantum computer is available. In this section some basic ideas of quantum cryptography are introduced. More specifically, a quantum analog of the Diffie-Hellman key exchange/distribution system, proposed by Bennett and Brassard in [7], will be addressed.

First let us define four *polarizations* as follows:

$$\{0^\circ,\ 45^\circ,\ 90^\circ,\ 135^\circ\} \overset{\text{def}}{=} \{\rightarrow,\ \nearrow,\ \uparrow,\ \nwarrow\}. \tag{5.1}$$

The quantum system consists of a transmitter, a receiver, and a quantum channel through which polarized photons can be sent [8]. By the law of quantum mechanics, the receiver can either distinguish between the *rectilinear polarizations* $\{\rightarrow,\ \uparrow\}$, or reconfigure to discriminate between the diagonal polarizations $\{\nearrow,\ \nwarrow\}$, but in any case, he cannot distinguish both types. The system works in the following way:

1. Alice uses the transmitter to send Bob a sequence of photons, each of them should be in one of the four polarizations $\{\rightarrow,\ \nearrow,\ \uparrow,\ \nwarrow\}$. For instance, Alice could choose, at random, the following photons:

$$\begin{array}{ccccccccc} \uparrow & \nearrow & \rightarrow & \nwarrow & \rightarrow & \rightarrow & \nearrow & \uparrow & \uparrow \end{array}$$

to be sent to Bob.

2. Bob then uses the receiver to measure the polarizations. For each photon received from Alice, Bob chooses, at random, the following type of measurements $\{+, \ \times\}$:

$$\begin{array}{ccccccccc} + & + & \times & \times & + & \times & \times & \times & + \end{array}$$

3. Bob records the result of his measurements but keeps it secret:

$$\begin{array}{ccccccccc} \uparrow & \rightarrow & \nearrow & \nwarrow & \rightarrow & \nearrow & \nearrow & \nearrow & \uparrow \end{array}$$

4. Bob publicly announces the type of measurements he made, and Alice tells him which measurements were of correct type:

$$\begin{array}{ccccccccc} \surd & & & \surd & \surd & & \surd & & \surd \end{array}$$

5. Alice and Bob keep all cases in which Bob measured the correct type. These cases are then translated into bits $\{0, 1\}$ and thereby become the key:

$$\begin{array}{ccccccccc} \uparrow & & & \nwarrow & \rightarrow & & \nearrow & & \uparrow \\ \\ 1 & & & 1 & 0 & & 0 & & 1 \end{array}$$

6. Using this secret key formed by the quantum channel, Bob and Alice can now encrypt and send their ordinary messages via the classic public-key channel.

An eavesdropper is free to try to measure the photons in the quantum channel, but, according to the law of quantum mechanics, he cannot in general do this without disturbing them, and hence, the key formed by the quantum channel is secure.

## Problems for Sect. 5.4

1. Explain what are the main features of quantum cryptography?
2. Explain why the quantum key distribution is quantum-computing resistant?
3. Use the idea explained in this section to simulate the quantum key distribution and to generate a string of 56 characters for a DES key.
4. Use the idea explained in this section to simulate the quantum key distribution and to generate a stream of 128 or 256 characters for an AES key.

## 5.5 DNA Biological Cryptography

The world was shocked by a paper [1] of Adleman (the "A" in the RSA), who demonstrated that an instance of the NP-complete problem, more specifically, the Hamiltonian path problem (HPP), can be solved in polynomial time on a DNA biological computer (for more information on biological computing, see, e.g., [2] and [33]. The fundamental idea of DNA-based biological computation is that of a set of DNA strands. Since the set of DNA strands is usually kept in a test tube, the test tube is just a collection of pieces of DNA. In what follows, we shall first give a brief introduction to the DNA biological computation.

**Definition 5.1.** A *test tube* (or just tube for short) is a set of molecules of DNA (i.e., a multi-set of finite strings over the alphabet $\Sigma = \{A, C, G, T\}$). Given a tube, one can perform the following four elementary biological operations:

(1) **Separate** or **Extract**: Given a tube $T$ and a string of symbols $S \in \Sigma$, produce two tubes $+(T, S)$ and $-(T, S)$, where $+(T, S)$ is all the molecules of DNA in $T$ which contain the consecutive subsequence $S$ and $-(T, S)$ is all of the molecules of DNA in $T$ which do not contain the consecutive sequence $S$.
(2) **Merge**: Given tubes $T_1, T_2$, produce the multi-set union $\cup(T_1, T_2)$:

$$\cup\,(T_1, T_2) = T_1 \cup T_2 \tag{5.2}$$

(3) **Detect**: Given a tube $T$, output "yes" if $T$ contains at least one DNA molecule (sequence) and output "no" if it contains none.
(4) **Amplify**: Given a tube $T$, produce two tubes $T'(T)$ and $T''(T)$ such that

$$T = T'(T) = T''(T). \tag{5.3}$$

Thus, we can replicate all the DNA molecules from the test tube.

These operations are then used to write "programs" which receive a tube as input and return either "yes" or "no" or a set of tubes.

**Example 5.1.** Consider the following program:

(1) Input(T)
(2) $T_1 = -(T, C)$
(3) $T_2 = -(T_1, G)$
(4) $T_3 = -(T_2, T)$
(5) Output(Detect($T_3$))

The model defined above is an unrestricted one. We now present a restricted biological computation model:

**Definition 5.2.** A tube is a multi-set of aggregates over an alphabet $\Sigma$ which is not necessarily $\{A, C, G, T\}$. (An aggregate is a subset of symbols over $\Sigma$.) Given a tube, there are three operations:

(1) **Separate**: Given a tube $T$ and a symbol $s \in \sum$, produce two tubes $+(T, s)$ and $-(T, s)$, where $+(T, s)$ is all the aggregates of $T$ which contain the symbols $s$ and $-(T, s)$ is all of the aggregates of $T$ which do not contain the symbol $s$.

(2) **Merge**: Given tube $T_1, T_2$, produce

$$\cup\,(T_1, T_2) = T_1 \cup T_2 \tag{5.4}$$

(3) **Detect**: Given a tube $T$, output "yes" if $T$ contains at least one aggregate or output "no" if it contains none.

**Example 5.2.** (3-colourability problem) Given an $n$ vertex graph $G$ with edges $e_1, e_2, \cdots, e_z$, let

$$\Sigma = \{r_1, b_1, g_1, r_2, b_2, g_2, \cdots, r_n, b_n, g_n\}.$$

and consider the following restricted program on input

$$\begin{aligned} T = \{\alpha \mid\ & \alpha \subseteq \Sigma, \\ & \alpha = \{c_1, c_2, \cdots, c_n\}, \\ & [c_i = r_i \textit{ or } c_i = b_i \textit{ or } c_i = g_i], i = 1, 2, \cdots, n\} \end{aligned}$$

```
(1) Input(T).
(2) for $k = 1$ to $z$. Let $e_k = \langle i, j \rangle$:
    (a) $T_{\text{red}} = +(T, r_i)$ and $T_{\text{blue or green}} = -(T, r_i)$.
    (b) $T_{\text{blue}} = +(T_{\text{blue or green}}, b_i)$ and $T_{\text{green}} = -(T_{\text{blue or green}}, b_i)$.
    (c) $T_{\text{red}}^{\text{good}} = -(T_{\text{red}}, r_j)$.
    (d) $T_{\text{blue}}^{\text{good}} = -(T_{\text{blue}}, b_j)$.
    (e) $T_{\text{green}}^{\text{good}} = -(T_{\text{green}}, g_j)$.
    (f) $T' = \cup(T_{\text{red}}^{\text{good}}, T_{\text{blue}}^{\text{good}})$.
    (g) $T = \cup(T_{\text{green}}^{\text{good}}, T')$.
(3) Output(Detect(T)).
```

**Theorem 5.2.** (Lipton, 1994) Any SAT problem in $n$ variables and $m$ clauses can be solved with at most $\mathcal{O}(m + 1)$ separations, $\mathcal{O}(m)$ merges, and one detection.

The above theorem implies that biological computation can be used to solve all problems in $\mathcal{NP}$, although it does not mean all instances of $\mathcal{NP}$ can be solved in a feasible way. From a computability point of view, neither the quantum computation model nor the biological computation model has more computational power than the Turing machine. Thus, we have an analogue of Church-Turing thesis for quantum and biological computations:

**Quantum and Biological Computation Thesis**: An arithmetic function is computable or a decision problem is decidable by a quantum computer or by a biological computer if and only if it is computable or decidable by a Turing machine.

This means that from a complexity point of view, both the quantum computation model and the biological computation model do have some more computational power than the Turing machine. More specifically, we have the following complexity results about quantum and biological computations:

1. Integer factorization and discrete logarithm problems are believed to be intractable in Turing machines; no efficient algorithms have been found for these two classical, number-theoretic problems; in fact, the best algorithms for these two problems have the worst-case complexity $\Theta\left((\log n)^2(\log\log n)(\log\log\log n)\right)$. But, however, both of these two problems can be solved in polynomial time by quantum computers.
2. The famous Boolean formula satisfaction problem (SAT) and directed HPP are proved to be $\mathcal{NP}$-complete, but these problems, and in fact any other $\mathcal{NP}$-complete problems, can be solved in polynomial biological steps by biological computers.

Now we are in a position to discuss the DNA-based cryptography [23]. We first study a DNA analog of one-time pad (OTP) encryption; its idea may be described as follows:

1. **Plaintext encoding**: The plaintext: $M$ is encoded in DNA strands.
2. **Key generation**: Assemble a large OTP in the form of DNA strands.
3. **OTP substitution**: Generate a table that randomly maps all possible strings of $M \rightarrow C$ such that there is a unique reverse mapping $M \leftarrow C$.
4. **Encryption**: Substitute each block of $M$ with the ciphertext $C$ given by the table to get $M \rightarrow C$.
5. **Decryption**: Reverse the substitutions to get $C \rightarrow M$.

The DNA implementation of the above scheme may be as follows:

1. **Plaintext in DNA**: Set one test tube of short DNA strands for $M$.
2. **Ciphertext in DNA**: Set another test tube of different short DNA strands for $C$.
3. **Key generation**: Assemble a large OTP in the form of DNA strands.
4. **OTP substitution**: Maps $M$ to $C$ in a random yet reversible way.
5. **Encryption : DNA substitution OTDs**: Use long DNA OTPs containing many segments; each contains a cipher word followed by a plaintext word. These word-pair DNA strands are used as a lookup table in conversion of plaintext into ciphertext for $M \rightarrow C$.

6. **Decryption**: Just do the opposite operation to the previous step for $C \rightarrow M$.

Just the same as stream cipher, we could use the operation XOR, denoted by $\oplus$, to implement the DNA OTP encryption as follows:

1. **DNA plaintext test tube**: Set one test tube of short DNA strands for $M$.
2. **DNA ciphertext test tube**: Set another test tube of different short DNA strands for $C$.
3. **Key generation**: Assemble a large OTP in the form of DNA strands.
4. **Encryption**: Perform $M \oplus$OTPs to get cipher strands; remove plaintext strands.
5. **Decryption**: Perform $C \oplus$ OTPs to get back plaintext strands.

**Problems for Sect. 5.5**

1. Explain how DNA computing can be used to solve the HPP.
2. Explain what are the main features of DNA biological cryptography?
3. Explain why DNA biological cryptography is a quantum-computing resistant?
4. DNA molecular biologic cryptography, e.g., Reif's OTP DNA cryptosystem developed in [23], is a new development in cryptography. Give a complete description and critical analysis of the Reif's DNA-based OTPs.
5. Write an assay to compare the main features of the classic, the quantum, and the DNA cryptography.

## 5.6 Conclusions, Notes, and Further Reading

Quantum-computing resistant, or quantum-attack resistant, or just quantum resistant cryptography is an important research direction in modern cryptography, since once a practical quantum computer can be build, all the public-key cryptography based on IFP, DLP, and ECDLP can be broken in polynomial time. As Bill Gates noted in his book [22]:

> We have to ensure that if any particular encryption technique proves fallible, there is a way to make an immediate transition to an alternative technique.

We need to have quantum resistant cryptographic systems ready at hand, so that we can use these cryptosystems to replace these quantum attackable cryptosystems. In this chapter, we only discussed some quantum resistant cryptographic systems, including quantum cryptography; interested readers should consult the following references for more information: [5, 6, 8, 9, 12, 13, 15, 18, 19, 21, 28–30, 34, 35, 37, 38, 42–44, 46, 52–54, 56, 57, 60, 61]. Note that in literatures, quantum-computing resistant cryptography is also called *post-quantum cryptography*. Springer publishes the proceedings of the post-quantum cryptography conferences [10, 16, 49, 62].

Just the same as quantum computing and quantum cryptography, DNA molecular computation is another type of promising computing paradigm and cryptographic scheme. Unlike the traditional computing model, DNA molecular computing is analog, not digital, so it opens a completely different phenomena to solve the hard computational problem. As can be seen from our above discussion, DNA computing has the potential to solve the NP-completeness problems such as the famous HPP and the satisfiability problem (SAT). Of course there is a long way to go to truly build up a practical DNA computer. Reader may consult the following references for more information on DNA computing and cryptography: [3, 4, 11, 14, 24, 25, 31, 36, 47, 48, 50, 55, 58].

Chaos-based cryptography [41, 51, 59] may be another type of good candidate for quantum resistant cryptography; readers are suggested to consult [32] for more information. Yet, there are another candidates for quantum resistant cryptography based on the conjectured difficulty of finding isogenies between supersingular elliptic curves [30], since the fastest known quantum algorithms for constructing isogenies between supersingular elliptic curves is exponential (however, the construction of isogenies between ordinary elliptic curves can be done in subexponential time).

## REFERENCES

[1] L.M. Adleman, Molecular computation of solutions to combinatorial problems. Science **266**, 1021–1024 (1994)

[2] L.M. Adleman, On constructing a molecular computer, in *DNA Based Computers*, ed. by R. Lipton, E. Baum (American Mathematical Society, Providence, 1996), pp. 1–21

[3] R.D. Barish, P. Rothemund, E. Winfree, Two computational primitives for algorithmic self-assembly: copying and counting. Nano Lett. **5**(12), 2586–2592 (2005)

[4] Y. Benenson, B. Gill, U. Ben-Dor et al., An autonomous moleular computer for logical control of gene expressions. Nature **429**, 6990, 423–429 (2004)

[5] C.H. Bennett, Quantum cryptography using any two nonorthogonal sates. Phys. Rev. Lett. **68**, 3121–3124 (1992)

[6] C.H. Bennett, Quantum information and computation. Phys. Today **48**(10), 24–30 (1995)

[7] C.H. Bennett, G. Brassard, Quantum cryptography: public key distribution and coin tossing, in *Proceedings of the IEEE International Conference on Computers Systems and Singnal Processing* (IEEE, New York, 1984), pp. 175–179

[8] C.H. Bennett, G. Brassard, A.K. Ekert, Quantum cryptography. Sci. Am. 26–33 (1992)

[9] E.R. Berlekampe, R.J. McEliece, H. van Tilburg, On the inherent intractability of certain coding problems. IEEE Trans. Inf. Theor. **IT-24**, 384–386 (1978)

[10] D.J. Bernstein, J. Buchmann, E. Dahmen (eds.), *Post-Quantum Cryptography* (Springer, Berlin, 2010)

[11] D. Boneh, C. Dunworth, R. Lipton et al., On the computational power of DNA. Discrete Appl. Math. **71**(1), 79–94 (1996)

[12] G. Brassard, Quantum computing: the end of classical cryptography? ACM SIGACT News **25**(3), 13–24 (1994)

[13] G. Brassard, C. Crépeau, 25 years of quantum cryptography. ACM SIGACT News **27**(4), 15–21 (1996)

[14] D. Bray, Pretein molecular as computational elements in living cells. Nature **376**, 6538, 307–312 (1995)

[15] D. Bruss, G. Erdélyi, T. Meyer, T. Riege, J. Rothe, Quantum cryptography: a survey. ACM Comput. Surv. **39**(2), Article 6, 1–27 (2007)

[16] J. Buchmann, J. Ding (eds.), in *Post-Quantum Cryptography*. Lecture Notes in Computer Science, vol. 5299 (Springer, Berlin, 2008)

[17] J.Y. Cai, T.W. Cusick, A lattice-based public-key cryptosystem. Inf. Comput. **151**(1–2), 17–31 (1999)

[18] E.F. Canteaut, N. Sendrier, Cryptanalysis of the original McEliece cryptosystem, in *Advances in Cryptology – AsiaCrypto'98*. Lecture Notes in Computer Science, vol. 1514 (Springer, Berlin, 1989), pp. 187–199

[19] P.-L. Cayrel, M. Meziani, Post-quantum cryptography: code-based signatures, in *Advances in Computer Science and Information Technology*. Lecture Notes in Computer Science, vol. 6059 (Springer, Berlin, 2010), pp. 82–99

[20] J. Ding, J.E. Gower, D.S. Schmidt, *Multivariate Public Key Cryptosystems* (Springer, Berlin, 2006)

[21] H. Dinh, C. Moore, A, Russell, McEliece and Niederreiter cryptosystems that resist quantum fourier sampling attacks, in *Advances in Cryptology – Crypto 2011*. Lecture Notes in Computer Science, vol. 6841 (Springer, Berlin, 2011), pp. 761–779

[22] B. Gates, *The Road Ahead* (Viking, New York, 1995)

[23] A. Gehani, T.H. LaBean, J.H. Reif, DNA-based cryptography, in *Molecular Computing*. Lecture Notes in Computer Science, vol. 2950 (Springer, Berlin, 2004), pp. 167–188

[24] T. Gramb, A. Bornholdt, M. Grob et al., *Non-Standard Computation* (Wiley-VCH, Weinheim, 1998)

[25] M. Guo, M. Ho, W.L. Chang, Fast parallel molecular solution to the dominating-set problem on massively parallel bio-computing. Parallel Comput. **30**, 1109–1125 (2004)

[26] J. Hoffstein, J. Pipher, J.H. Silverman, A ring-based public-key cryptosystem, in *Algorithmic Number Theory ANTS-III*. Lecture Notes in Computer Science, vol. 1423 (Springer, Berlin, 1998), pp. 267–288

[27] J. Hoffstein, N. Howgrave-Graham, J. Pipher, J.H. Silverman, W. Whyte, NTRUEncrypt and NTRUSign: efficient public key Algorithmd for a post-quantum world, in *Proceedings of the International Workshop on Post-Quantum Cryptography (PQCrypto 2006)*, (Springer, Berlin, 2006), pp. 71–77

[28] R.J. Hughes, Cryptography, quantum computation and trapped ions. Phil. Trans. R. Soc. Lond. Ser. A **356**, 1853–1868 (1998)

[29] H. Inamori, in *A Minimal Introduction to Quantum Key Distribution*. Centre for Quantum Computation, Clarendon Laboratory (Oxford University, Oxford, 1999)

[30] D. Jao, L. De Feo, Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies, in *Post-Quantum Cryptography*, ed. by Yang. Lecture Notes in Computer Science, vol. 7071 (Springer, Berlin, 2011), pp. 19–34

[31] N. Jonoska, G. Paun, G. Rozenberg (eds.), in *Molecular Computing*. Lecture Notes in Computer Science, vol. 2950 (Springer, Berlin, 2004)

[32] L. Kocarev, S. Lian, *Chaos-Based Cryptography* (Springer, Berlin, 2011)

[33] E. Lamm, R. Unger, *Biological Computation* (CRC Press, Boca Raton, 2011)

[34] A.K. Lenstra, H.W. Lenstra Jr., L. Lovász, Factoring polynomials with rational coefficients. Math. Ann. **261**, 515–534 (1982)

[35] H.W. Lenstra Jr., Lattices, in *Algorithmic Number Theory*, ed. by J.P. Buhler, P. Stevenhagen (Cambridge University Press, Cambridge, 2008), pp. 127–182

[36] R.Lipton, DNA solution of hard computational problems. Science **268**, 5210, 542–545 (1995)

[37] H.K. Lo, Quantum cryptography, in *Introduction to Quantum Computation and Information*, ed. by H.K. Lo, S. Popescu, T. Spiller (World Scientific, Singapore, 1998), pp. 76–119

[38] H. Lo, H. Chau, Unconditional security of quantum key distribution over arbitrary long distances. Science **283**, 2050–2056 (1999)

[39] F.J. MacWilliams, N.J.A. Sloana, *The Theory of Error Correcting Codes* (North-Holland, Amsterdam, 2001)

[40] R.J. McEliece, A Public-Key Cryptosystem based on Algebraic Coding Theory, pp. 583–584. JPL DSN Progress Report 42–44 (1978)

[41] I. MishkovskiK, L. Kocarev, Chaos-based public-key cryptography, in *Chaos-Based Cryptography*, ed. by L. Kocarev, S. Lian (Springer, Berlin, 2011), pp. 27–66

[42] P.Q. Nguyen, B. Vallée, *The LLL Algorithm: Survey and Applications* (Springer, Berlin, 2011)

[43] H. Niederreiter, Knapsack type cryptosystems and algebraic coding theory. Probl. Contr. Inf. Theor. **15**, 159–166 (1986)

[44] M.A. Nielson, I.L. Chuang, *Quantum Computation and Quantum Information*, 10th Anniversary edn. (Cambridge University Press, Cambridge, 2010)

[45] Y. Pan, Y. Deng, Cryptanalysis of the Cai-Cusick lattice-based public-key cryptosystem. IEEE Trans. Inf. Theor. **57**(3), 1780–1785 (2011)

[46] R.A. Perlner, D.A. Cooper, Quantum resistant public key cryptography, in *Proceedings of the 8th Symposium on Identity and Trust on the Internet*, Gaithersburg, MD, 14–16 April (ACM, New York, 2009), pp. 85–93

[47] C. Popovici, Aspects of DNA cryptography, Ann. Univ. Craiova, Math. Comput. Sci. Ser **37**(3), 147–151 (2010)

[48] J.H. Reif, Parallel biomolecular computation. Algorithmica **25**, 142–175 (1999)

[49] N. Sendrier (ed.), in *Post-Quantum Cryptography*. Lecture Notes in Computer Science, vol. 6061 (Springer, Berlin, 2010)

[50] H. Singh, K. Chugh, H. Dhaka, A.K. Verma, DNA-based cryptography: an approach to secure mobile networks. Int. J. Comp. Appl. **1**(19), 82–85 (2010)

[51] E. Solak, Cryptanalysis of chaotic ciphers, in *Chaos-Based Cryptography*, ed. by L. Kocarev, S. Lian (Springer, Berlin, 2011), pp. 227–254

[52] W. Trappe, L. Washington, *Introduction to Cryptography with Coding Theory*, 2nd edn. (Prentice-Hall, Englewood Cliffs, 2006)

[53] H. van Tilborg (ed.), *Encyclopedia of Cryptography and Security* (Springer, Berlin, 2005)

[54] H. van Tilburg, On the McEliece public-key cryptography, in *Advances in Cryptology – Crypto'88*. Lecture Notes in Computer Science, vol. 403 (Springer, Berlin, 1989), pp. 119–131

[55] R. Unger, J. Moult, Towards computing with protein. Proteine **63**, 53–64 (2006)

[56] J.L. Walker, *Codes and Curves* (American Mathematical Society and Institute for Advanced Study, Providence, 2000)

[57] C.P. Williams, *Explorations in Quantum Computation*, 2nd edn. (Springer, New York, 2011)

[58] E. Winfree, F. Liu, L.A. Wenzler et al., Design and self-assembly of two-dimensional DNA crystals. Nature **394**, 6693, 539–544 (1998)

[59] D. Xiao, X. Liao, S. Deng, Chaos-based Hash function, in *Chaos-Based Cryptography*, ed. by L. Kocarev, S. Lian (Springer, Berlin, 2011), pp. 137–204

[60] S.Y. Yan, *Cryptanalyic Attacks on RSA* (Springer, New York, 2009)

[61] S.Y. Yan, *Primality Testing and Integer Factorization in Public-Key Cryptography*, 2nd Edition (Springer, New York, 2010)

[62] B. Yang (ed.), in *Post-Quantum Cryptography*. Lecture Notes in Computer Science, vol. 7071 (Springer, New York, 2011)