Quantum-Robust Public Key Methods

TBC, Edinburgh Napier University, UK.

Abstract

Recent years have seen a substantial amount of research into quantum computers; these machines use a different and non-classic paradigm for computation, specifically the quantum mechanical phenomena. It has been shown that they will have the ability to solve mathematical problems that conventional computers find intractable, however the threat that quantum computing brings differs significantly on each specific primitive. In the event that industrial quantum computers are built, it is known that it will suffice to double the secret key sizes of symmetric encryption algorithms to defend against the cryptanalytic speedups that quantum computers will bring. However, the public-key cryptographic systems in use today will be insecure and the confidentiality and integrity of digital communications will be compromised. This threat cannot be ignored and has led to a global effort in the standardisation of quantum-robust public-key cryptographic systems.

Keywords: Post-quantum cryptography, Lattice, LWE, Ring-LWE

1. Introduction

The digital age has seen enormous growth in global communication and public-key cryptography has become the back-bone for this. Providing both confidentiality and integrity of data, public-key cryptography has provided secure communication and has become a fundamental piece of information security. The ability for individuals, businesses and governments to communicate securely is absolutely critical in todays connected world.

Three main cryptographic functionalities underpin the majority of communication protocols. These are: public key encryption, key exchange, and digital signatures. There are different methods of implementing these, however the commonly used cryptographic systems are: elliptic curve, RSA (Rivest-Shamir-Adleman) and Diffie-Hellman key exchange. These rely on the computational difficulty of certain number theoretic problems, namely the Integer Factorization Problem

(IFP) or the Discrete Log Problem (DLP) over various groups. Currently, the fastest known algorithm for Integer Factorization is the general number field sieve, which runs in sub-exponential time [1]. However, in 1994 Peter Shor of Bell Laboratories developed a quantum algorithm for integer factorization that runs in polynomial time [2]. The consequences of this are that all public key cryptosystems based on the IFP will be insecure when a large-scale quantum computer is constructed.

Since Shor's discovery in 1994, there has been significant development into the theory of quantum algorithms. This development has led to the discovery of algorithms capable of achieving exponential speedup with quantum computation. It has also been shown that an exponential speedup is impossible on search algorithms, providing confidence that symmetric algorithms and hash functions should be usable in a quantum era [3]. However, this does not mean that they will not be affected, speedups for wide-ranging classes of problems related to collision finding and searching have been discovered. One particular proposal, Grover's algorithm, details a near-quadratic speedup on unstructured search problems [4]. This algorithm can be applied to any problem within the complexity class NP; the NP class has an informal definition as a problem that has a solution which can be checked by a deterministic classical computer in polynomial time [4]. The consequences of the developed speedups are that larger key sizes will be required, especially in symmetric key cryptography where it is believed that doubling the key size will be sufficient to preserve security [5].

The National Institutes of Science and Technology (NIST) have recognised the risk that quantum-computers brings to secure communication and has set out to standardise quantum-robust cryptographic systems. The aspiration is to allow experts from academic institutions and industry to work together to identity the strengths and weaknesses of a variety of options, and then work towards standardising those deemed suitable.

2. Factoring Based Cryptography

The first and most widely used public-key cryptosystem is RSA, named after its co-founders Rivest-Shamir-Adleman, which relies upon the difficulty of the integer factoring problem (IFP) [6]. There are numerous methods and algorithms for integer factorisation which are categorised into either:

· SPECIAL-PURPOSE FACTORING ALGORITHMS - whereby the running time depends on the size of the factor found. Examples include: Trial division [7], Pollard's ρ method [8], or algebraic-group factorisation algorithms such as Pollard's $\rho - 1$ [9], and Lenstra elliptic curve factorisation [10].

· GENERAL-PURPOSE FACTORING ALGORITHMS - whereby the running time depends on the size of the number to be factored. Examples include the continued fraction (CFRAC) method [11], quadratic sieve [12] and the number field sieve (NFS).

Currently, the largest known RSA number to be factorised is 768-bits [13]. This was achieved in 2009 over the span of two years and was completed by a collection of parallel computers, the CPU-time required for the factorisation is the equivalent of 2000 years computing on a single-core 2.2Ghz processor. The bulk of the work in this factorisation was completed using a lattice sieving method.

From a computational complexity point of view, the IFP is still an infeasible problem as all algorithms run in sub-exponential time or higher. There is one exception to this, which is the quantum-factoring algorithm proposed by Shor [2]. He showed that IFP can be solved in BQP, where BQP is the class of problem that can be solved by a quantum computer in polynomial time [4]. Hence, all cryptographic systems that are based on the IFP can be broken in polynomial-time on a quantum computer. Shor presented the quantum order finding algorithm, the quantum factoring algorithm and methods to extend these algorithms to break RSA [14].

The current development into quantum computing has led to the quantum factorisation of:

· 143 using Shors algorithm which required a minimum of 8 qubits [15].

· 56153 using a minimization technique, using only 4 qubits [16].

[16] also demonstrates how to factor up to 291311 with 6 qubits, however this technique only works with numbers that have factors that are divisible by two, which is not opportune unless the attacker knows that the numbers will factors this way.

3. Logarithm Based Cryptography

The difficulty of discrete logarithms means that even supercomputers today struggle and it is for this reason that the Discrete Logarithm Problem (DLP) is used for cryptographic schemes. DLP can be applied to various groups, with some being easier to solve than others [17].

· THE ADDITIVE GROUP $\mathbb{F}_p^+$ is easy to compute as it only requires the multiplicative inverse of x in $\mathbb{F}_p$, which can be computed in polynomial-time using the Euclidean algorithm..

· THE MULTIPLICATIVE GROUP $\mathbb{F}_p^*$ is considered generally hard to compute. There are no known polynomial-time algorithms to solve it and the most powerful known method, the Index calculus method [18], is a sub-exponential algorithm.

· AN ELLIPTIC CURVE $E(\mathbb{F}_q)$ is also hard to compute. The best known algorithm to solve being Pollard's ρ method, which solves Elliptic Curve Discrete Log Problem (ECDLP) in exponential time.

The security of the Diffie Hellman key exchange algorithm relies upon the DLP in $\mathbb{F}_p^*$ and the ElGamal public-key cryptosystem is based on the same primary problem. Koblitz and Miller [19][20] presented a different approach and replaced the finite field Fq with an elliptic curve, E. They believed that DLP in the elliptic curve group $E(\mathbb{F}_q)$ might be harder to solve than in the multiplicative group $(\mathbb{F}_q^*)$, which ultimately led to the birth of elliptic curve cryptography.

There are many classical methods to solve DLP which, similar to IFP, has been utilised to create cryptographic public-key schemes and protocols. Examples include: Pollard's ρ method [21], Pohlig-Hellmans method [22], index calculus method [18]. ECDLP is similar to DLP in that it is an inverse problem, in this circumstance it is trying to find the additive inverse of $E(\mathbb{F}_q)$. The method for solving this problem is an elliptic curve variation of the Euclidean algorithm. Koblitz and Miller were correct in their theorum as the computation of ECDLP is more difficult than DLP and the fastest general-purpose algorithm known for solving ECDLP is Pollard's ρ method.

Shor showed that DLP could be solved in BQP, so similar to IFP, all cryptographic systems based on DLP can be broken by a quantum computer. The quantum algorithm for solving DLP was proposed in 1994 by Shor [14] and is also applicable

to ECDLP. Since this discovery, cryptographers have been researching ways to increase the efficiency of Shor's algorithm with notable quantum algorithms including Proos and Zalka [23]. They proposed a quantum algorithm that solves ECDLP over the finite field $\mathbb{F}_p$, however in practice, elliptic curves use a finite field $\mathbb{F}_{2^m}$. Kaye and Zalka extended the Proos-Zalka algorithm [24] to make it applicable to the finite field $\mathbb{F}_{2^m}$ more specifically, they use the Euclid's algorithm for polynomials to computer inverses in $\mathbb{F}_{2^m}$.

Interestingly, [23] showed that an ECDLP-based cryptographic system could be broken with a smaller quantum computer than an IFP-based cryptographic system with the same level of security. More specifically, that a "160-bit ECC key could be broken on a quantum computer using 1000 qubits, whereas factoring the security equivalent 1024-bit RSA modulus would require 2000 qubits". This reverses the situation seen within classic computation, where ECC uses a noticeably smaller key size to provide the same level of security as RSA. These shorter key sizes will still prove to be useful in wireless security, where the key size is limited.

4. Standardisation Process

Understanding that quantum-computers will be capable of solving the infeasible IFP, DLP and ECDLP in polynomial-time, means that public-key cryptosystems based on these problems (such as RSA, DH and ECC) will be broken in polynomial time. Despite this, it is incorrect to assume that quantum computers will be able to break all cryptographic systems and protocols. Other infeasible problems exist, such as the shortest lattice problem, where a quantum computer would have the computational power equal to that of a classic computer. In fact, quantum computers have not been shown to solve any NP-complete problems so far.

Quantum computers can speed up the computation of any periodic function. The Fast Fourier Transform (FFT), which can be used to compute the period of a function, can be extended into a Quantum Fourier Transform (QFT) [25]. This means that any computation that FFT cannot be applied too, is potentially quantum-robust.

Although a formidable quantum computer has not yet been produced, the research into quantum computing is progressing, a fact demonstrated by the increase in size of numbers being factorised. "It is estimated that there is a 1/7 chance that RSA-2048 will be broken by 2026, and a 1/2 chance that it will be broken by

2031" [26]. This, along with the time it takes to develop and test a cryptographic system means that the need to locate and standardise new cryptographic systems has never been more apparent. Armed with the knowledge that there are infeasible problems that a quantum computer has not been shown to break, cryptographers around the world are creating, documenting and testing new cryptographic systems based. In 2016, the National Institute of Standards and Technology (NIST) kicked off the standardisation process for post-quantum cryptographic systems. Their aspiration is to standardise multiple options, whilst detailing the strengths and weaknesses of each. This progressed to the second round of candidates in January 2019 with 17 key encapsulation methods (KEM) and 9 signature submissions [27]. For the KEM submissions: 9 are lattice-based, 7 code-based and 1 uses SIDH. For signature submissions: 4 multivariate, 3 lattice-based, 1 hash-based and 1 'other'. This paper looks further into the lattice-based public-key cryptographic systems.

5. Lattice-based Cryptography

Cryptography based on ring properties and particularly lattice reduction is a promising direction of post-quantum cryptography [28][29]. The lattice reduction is a reasonably well-studied hard problem that is currently not known to be solved in polynomial-time, or even in sub-exponential time, on a quantum computer. Lattice-based cryptographic constructions enjoy strong security proofs based on worst-case hardness, relatively efficient implementations, as well as great simplicity.

A lattice is a set of points in n-dimensional space with a periodic structure. As mentioned previously, periodic functions can be computed with the Fourier transform which can further be extended into quantum algorithms, however there have not yet been any quantum algorithms that can solve lattice problems. It appears that the periodicity finding technique, which is used in Shor's factoring algorithm and related quantum algorithms, does not seem to be applicable to lattice problems. Micciancio and Regev conclude that "there is no polynomial time quantum algorithm that approximates lattice problems to within polynomial factors" [30].

A lattice is defined as the set of all integer combinations of n-linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ in $\mathbb{R}_n$. The basis of the lattice is the set of vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ [30].

5.1. Problems

The three main computational problems on lattices are as follows:

· SHORTEST VECTOR PROBLEM (SVP) : Given a lattice basis $\mathbf{B}$, find the shortest nonzero vector in $\mathscr{L}(\mathbf{B})$.

· CLOSEST VECTOR PROBLEM (CVP) : Given a lattice basis $\mathbf{B}$ and a target vector t (not necessarily in the lattice), find the lattice point $v \in \mathscr{L}(\mathbf{B})$ closest to t.

· SHORTEST INDEPENDENT VECTORS PROBLEM (SIVP) : Given a lattice basis $\mathbf{B} \in \mathbb{Z}^{n \times n}$, find n-linearly independent lattice vectors $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_n]$ (where $\mathbf{s}_i \in \mathscr{L}(\mathbf{B})$ for all i) minimizing the quantity $||\mathbf{S}|| = \max_i ||\mathbf{s}_i||$.

Each of the above problem comes with different variants, their conjectured intractability is the foundation for a large number of cryptographic applications of lattices. Van Emde Boas showed that the lattice-reduction problem is classed as hard [31], Atjai then proved that SVP is NP-hard in L2 for randomised reductions, and that the corresponding decision problem is NP-complete [32].

The Lenstra, Lenstra Lovasz (LLL) algorithm shows that SVP can be solved in polynomial time using an exponential approximation factor $2^{\mathscr{O}(n)}$ [28]. Algorithms that solve SVP with poly(n) factors either run in $2^{\mathscr{O}(n)}$ and require exponential space [33] or in $2^{\mathscr{O}(n \log n)}$ and require only polynomial space [34].

There are two average-case problems [30] that enjoy worst-case hardness guarantee: the learning with errors (LWE) problem [35][36][37] and the small integer solution (SIS) problem [30].

· LWE : Regev defined LWE [35] and proved that, under a quantum reduction, it enjoyed similar worst-case hardness.

· SIS : Atjai [38] proposed SIS and showed that it was at least as hard as approximating worst-case lattice problems to within a polynomial factor in the lattice dimension.

LWE and SIS problems have formed the basis of several cryptographic schemes, as seen in [35][36][37][39][40][41][42][43][44].

In most cases, in order to solve the LWE problem, one has to recover a secret vector when given a sequence of approximate random linear equations on $\mathbf{s}$. The LWE

problem is usually used to build primitives such as CPA or CCA-secure public-key encryption, identity-based encryption (IBE), or fully-homomorphic encryption schemes [45]. LWE can be defined as a search problem (sLWE), where the task is to recover the secret vector s, or as a decision problem (dLWE) that asks to distinguish LWE samples from uniformly random samples [35]. There is an equivalence between search and decision whereby if the search LWE problem is easy, then the decision LWE problem is easy, and visa versa. Variations of the LWE problem that utilise the polynomial rings were defined and have become another topic of research [46][47]. The lattices underlying this problem are module lattices, as in NTRU [48][30], and its hardness can be related to the worst case hardness of finding short vectors in ideal lattices [49][50].

· sLWE [35] : The learning with errors problem, search version, $\text{sLWE}_{n,m,q,\chi}$ with n unknowns, $m \geq n$ samples, modulo q and with error distribution χ is as follows: for a random secret s uniformly chosen in $\mathbb{Z}_q^n$, and given m samples of the form (a, b = $\langle$s, a$\rangle$ + e mod q) where e $\xleftarrow{\$}$ χ and a is uniform in $\mathbb{Z}_q^n$, recover the secret vector **s**.

· dLWE [35] : The learning with errors problem, decisional version, $\text{dLWE}_{n,m,q,\chi}$ with n unknowns, $m \geq n$ samples, modulo q and with error distribution χ is as follows: for a random secret s uniformly chosen in $\mathbb{Z}_q^n$, and given m samples either all of the form (a, b = $\langle$s, a$\rangle$ + e mod q) where e $\xleftarrow{\$}$ χ, or from the uniform distribution (a, b) $\xleftarrow{\$}$ $U(\mathbb{Z}^n \times \mathbb{Z}_q)$, decide if the samples come from the former or the latter case.

· rLWE, DECISIONAL VERSION [35] : Let R denote the ring $Z[X]/(X^n + 1)$ for n a power of 2, and $\mathbb{R}_q$ the residue ring $\mathbb{R}/qR$. The ring learning with errors problem, decisional version, $\text{dRLWE}_{m,q,\chi}$, with m unknowns, $m \geq 1$ samples, modulo q and with error distribution χ is as follows: for a uniform random secret s $\xleftarrow{\$}$ $U(\mathbb{R}_q)$, and given m samples either all of the form (**a**,**b** = **a**· **s**+**e** mod q) where the coefficients of **e** are independently sampled following the distribution χ, or from the uniform distribution (a, b) $\xleftarrow{\$}$ $U(\mathbb{R}_q \times \mathbb{R}_q)$, decide if the samples come from the former or the latter case.

5.2. Key Encapsulation Mechanisms (KEMs) Key Agreement

The general KEM consists of four algorithms:

· SETUP() -> PP : Outputs a public parameter

· KeyGen(pp) -> (pk, sk) : A key generation algorithm that takes the public parameter, then outputs a public encapsulation key **pk** and secret decapsulation key **sk**

· Encaps(pp, pk) -> (c,k) : An encapsulation algorithm that outputs a ciphertext $c \in C$ and session key $k \in K$.

· Decaps(sk,c) -> k : A decapsulation algorithm that takes a decapsulation key **sk**, a ciphertext **c** and then outputs a session key k (or an error symbol)

A key exchange protocol is an interactive protocol carried out between two parties, Alice and Bob. The goal is to produce a session key that is indistinguishable from random.

In authenticated key exchange protocols, the adversary can be active and controls all communications between parties; the parties are assumed to have authentically distributed trusted long-term keys out of band prior to the protocol. On the other hand, in unauthenticated key exchange protocols, the adversary can be passive and only obtains transcripts of communications between honest parties.

Two security properties for KEMs are Indistinguishability under Chosen Plaintext Attack (**IND-CPA**), Indistinguishability under Chosen Chiphertext Attack (**IND-CCA**). These are used to compare the relative strengths of various notions of security for public-key encryption, more information about these can be found [51].

5.3. A simple LWE key agreement

This section details a simple LWE key agreement in an unauthenticated situation between Alice and Bob. In this circumstance, Alice will play the role of the server and Bob will the client.

· Step 1 : A shared public key is generated by both parties - **A** in $\mathbb{Z}_q^{n \times m}$

· Step 2 : Alice generates her secret values - **s**, **e** in $\mathbb{Z}_q^m$

· Step 3 : Alice calculates **b = As + e**

· Step 4 : Alice sends b to Bob

· Step 5 : Bob generates his **secret** values - **s'**, **e'** in $\mathbb{Z}_q^m$

· STEP 6 : Bob calculates **b' = s'A + e'**

· STEP 7 : Bob sends **b'** to Alice

From this, Alice can calculate her shared secret with **b's** = s'As + **e's** $\approx$ **s'As** and Bob can calculate his shared secret with **s'b** $\approx$ **s'As**. These shared secret values are only approximate and need rounding.

5.4. Reconciliation & Peikert's KEM

Using an error-reconciliation mechanism, one component of the ciphertext can be replaced by a more compact element. The scheme was suggested by Ding [52], and extended by Peikert [53]. It allows both Alice and Bob to derive the session key from an approximately agreed psuedorandom ring element. Peikert's [53] methodology for rounding and reconciliation is used in several cryptographic methods, such as NewHope [54].

5.5. NIST Candidates

As mentioned previously, there are several proposals that have progressed to the second round of the NIST standardisation process. Two of the proposed lattice-based key exchange methods are "NewHope" and "Frodo: removes the ring!". NewHope is a Ring-LWE key exchange with many optimizations and conjectured 200-bit quantum security. This can be compared to, or is even faster than, state-of-the-art ECDH that has 128-bit (non-quantum) security. Google has experimentally deployed NewHope+ECDH in Chrome canary and its own web servers [54]. Frodo is a plain-LWE key exchange method that uses a variety of optimizations. It is conjectured $\geq$ 128-bit quantum security, and currently operates about 10x slower than NewHope, but only $\approx$ 2x slower than ECDH.

6. Implementation

The following gives an outline of the Python code, with comments indicated by '#' explaining the purpose of each section of code.

```
#SERVER-SIDE SETUP -----
#The public parameter a is generated by gen_a, which takes a 32 byte array seed
seed = os.urandom(params.SEEDBYTES)
A = gen.gen_a(seed)

#SERVER-SIDE -----
#Alice generates two polynomials, a secret (s_A) and an error (e_A)
s_A = gen.get_noise()
e_A = gen.get_noise()
#Alice creates b_A, which is a polynomial created from A, e_A and s_A
b_A = p.polymul(A,s_A)%params.Q
```

```
b_A = np.floor(p.polydiv(s_A,hlpr)[1])
b_A = p.polyadd(b_A,e_A)%params.Q
#Here, it is possible to use a fast quasi-logarithmic algorithm to complete
#the polynomial multiplication. Such as Number Theoretic Transform (NTT),
#Karatsuba or Schoolbook multiplication

#CLIENT-SIDE KEYGEN ------
#Alice (Server-side) would transmit the public-key, or the used seed,
#which would allow Bob to generate his own A value
A = gen.gen_a(seed)

#CLIENT-SIDE ------
#Bob generates his own error polynomial e' and a secret polynomial s'
s_B = gen.get_noise()
e_B = gen.get_noise()

#Alice shares A with Bob
#Now Bob creates b_B which is a polynomial created from A, e_B and s_B
#b_B = A x s_B + e_B
b_B = p.polymul(A,s_B)%params.Q
b_B = np.floor(p.polydiv(s_B,hlpr)[1])
b_B = p.polyadd(b_B,e_B)%params.Q

#SHARED SECRET ------
#Alice takes Bob's value (b_B) and multiplies by s_A, then divides by x**n+1
sharedAlice = np.floor(p.polymul(s_A,b_B)%params.Q)
sharedAlice = np.floor(p.polydiv(sharedAlice,hlpr)[1])%params.Q

#Bob takes Alice's value (b_A) and multiplies by s_B and divies by x**n+1
sharedBob = np.floor(p.polymul(s_B,b_A)%params.Q)
sharedBob = np.floor(p.polydiv(sharedBob,hlpr)[1])%params.Q

#ERROR ROUNDING ------
#BOB ------
u = np.asarray([0] * params.N)
i = 0

while (i < len(u)):
        if (len(b_B) <= i): break;
        if (int(b_B[i]/(params.Q/4)) == 0): u[i] = 0
        elif (int(b_B[i]/(params.Q/2)) == 0): u[i] = 1
        elif (int(b_B[i]/(3*params.Q/4)) == 0): u[i] = 0
        elif (int(b_B[i]/(params.Q)) == 0): u[i] = 1
        else:
                print("error!␣(1)")
        i+=1
i = 0

while (i < len(u)):
        #Region 0 (0 --- q/4 and q/2 --- 3q/4)
        if (u[i] == 0):
                if (sharedBob[i] >= params.Q*0.125 and sharedBob[i] < params.Q*0.625):
                        sharedBob[i] = 1
                else:
                        sharedBob[i] = 0

        #Region 1 (q/4 --- q/2 and 3q/4 --- q)
        elif (u[i] == 1):
                if (sharedBob[i] >= params.Q*0.875 and sharedBob[i] < params.Q*0.375):
                        sharedBob[i] = 0
                else:
                        sharedBob[i] = 1
        else:
                print("error!␣(2)")
        i += 1

#ALICE ------
i = 0
```

```
while (i < len(u)):
        #Region 0 (0 ---- q/4 and q/2 ---- 3q/4)
        if (u[i] == 0):
                if (sharedAlice[i] >= params.Q*0.125 and sharedAlice[i] < params.Q*0.625):
                        sharedAlice[i] = 1
                else:
                        sharedAlice[i] = 0

        #Region 1 (q/4 ---- q/2 and 3q/4 ---- q)
        elif (u[i] == 1):
                if (sharedAlice[i] >= params.Q*0.875 and sharedAlice[i] < params.Q*0.375):
                        sharedAlice[i] = 0
                else:
                        sharedAlice[i] = 1
        else:
                print("error! (3)")
        i += 1

#FUNCTION DEFINITIONS
#gen_a uses the SHAKE128 hash function to expand the psuedorandom seed and
#defines a function that absorbs a byte array into the internal state of
#SHAKE128. Then another function is used to obtain pseudorandom data by
#squeezing the internal state.

def gen_a(seed):
    hashing_algorithm = hashlib.shake_128()
    hashing_algorithm.update(seed)
    # 2200 bytes from SHAKE-128 function is enough data to get 1024
    coefficients smaller than 5q
    shake_output = hashing_algorithm.digest(2200)
    output = []
    j = 0
    for i in range(0,params.N):
        coefficient = 5 * params.Q
        # Reject coefficients that are greater than or equal to 5q:
        while coefficient >= 5 * params.Q:
            coefficient = int.from_bytes(
                shake_output[j * 2 : j * 2 + 2], byteorder = 'little')
            print('j=' + str(j))
            j += 1
            if j * 2 >= len(shake_output):
                print('Error: Not enough data from SHAKE-128')
                exit(1)
        output.append(coefficient)
        print('chose ' + str(coefficient))
    return output

#get_noise is used to generate the secret and error values. Centered binomial
#distribution is used for noise distribution, using parameter k = 8 for the
#secret and error. In comparison to using a discrete Gaussian sampler, the
#centered binomial distribution is easier and does not require high-precision
#computations or large tables.

def get_noise():
    coeffs = []
    for i in range(0, params.N):
        t = int.from_bytes(os.urandom(4), byteorder='little')
        d = 0
        for j in range(0, 8):
            d += (t >> j) & 0x01010101
        a = ((d >> 8) & 0xff) + (d & 0xff)
        b = (d >> 24) + ((d >> 16) & 0xff)
        coeffs.append(a + params.Q - b)
    return coeffs
```

7. Testing and Evaluation

One of the main advantages of using R-LWE based cryptography rather than LWE based cryptography is the size of the keys used. The size of the keys used in R-LWE keys are $\approx$ square root of LWE keys [46]. This means that to satisfy 128-bits of security, a R-LWE cryptographic system would use a public key $\approx$ 7000 bits in length [55], rather than the 49 million bits that an LWE scheme of the same security would use. However, this is still much bigger than current public-key algorithms like RSA and Elliptic Curve Diffie-Hellman that operate at 128-bit level security with public key sizes of 3072-bits and 256-bits respectively.

In the implemented code, a different distribution is used to generate the secret and error values. It also uses SHAKE128 in the gen_a function. This hashing method provides 128-bit security level for both collision and pre-image resistance. Through implementing this, the speed of the implementation is increased. Using Python's numpy module, the public A value can be generated with:

```
hlpr = [1] + [0] * (n-1) + [1]

A = numpy.floor(numpy.random.random(size=(n))*q)%q
A = numpy.floor(p.polydiv(A, hlpr)[1])
```

This does not offer the same security as the SHAKE128 method, however, the speed difference is displayed in Table 1. This shows the outlines the average time taken (taken from 100 generations) for each element of the implemented R-LWE code with and without the utilisation of the security optimisations, the value for 'q' used is $2^{32} - 1$.

Table 1: Average Time with/without SHAKE128, with RSA and ECDH for comparison

Key Size	With SHAKE128	Without SHAKE128
256	0.01244757	0.00619996
512	0.02462202	0.00674577
1024	0.03905223	0.01758617
2048	ERROR	0.04189856
3072	ERROR	0.08652870
4096	ERROR	0.11928102
7680	ERROR	0.21860398
15360	ERROR	5.38205074

NIST Guidelines for public key sizes with equivalent security levels are shown

in Table 2 [56]. The average time to generate the public and private keys using RSA and ECC are also shown, 100 iterations were used to gather a mean result.

Table 2: Average Time taken for equivalent security RSA and ECC

Security (bits)	RSA (key size)	RSA (s)	ECC (key length)	ECC (s)
80	1024	0.07287715	160	0.05234763
112	2048	0.25100964	224	0.09109507
128	3072	0.71940293	256	0.12470847
192	7680	33.5771813	384	0.29928764
256	15360	190.018915	512	0.63584411

In the R-LWE implementation utilising the SHAKE128, it can be seen that running the code with an N value at 2048 or higher causes an error, the actual NewHope proposal uses specific N and Q values due to their use of Number Theoretic Transform (NTT) throughout [54]. The code used to generate the RSA keys triggered an error when attempting anything less than 1024 in size. ECDH it can be seen that as the key size increases, so does the time taken to implement. NewHope claims 233-bits post-quantum security with its performance analysis located in their specification document [57].

The code used to generate these key values can be found at at [58].

8. Conclusions

Compared with traditional theory-based cryptosystems, such as RSA and ECC, the lattice-based cryptographic systems offer a higher operating speed, smaller computation operation and smaller computation complexity. This is due to the required operations including simple linear operation, matrix-vector multiplication, modular addition and modular multiplication of small integers. The security of some lattice-based systems are provably secure under a worst-case hardness assumption, rather than on the average case. They do have disadvantages of course, which are the long public and secret keys, which limits their practical application. It is also difficult to accurately estimate their security with known cryptanalysis techniques. It is not yet known whether the RLWE problem is easier to solve than the LWE problem, it can be speculated that the inclusion of an additional ring structure could add subtle changes in the choice of error distribution.

There is research into quantum-robust cryptographic systems that are not based on

lattice-problems, such as code-based cryptosystems and those based on supersingular isogenies. Some of these are looking like promising candidates in the NIST standardisation process, but there is still much research to be done into all areas to find the strengths and weaknesses of each. This includes testing the efficiency, the security and other rationale. It is imperative that more than one cryptosystem is standardised as future development into quantum-mechanics could reveal further quantum algorithms that may make particular cryptosystems impotent.

This could also include research into other methods of computation, there is active research into DNA molecular cryptography and Chaos-based cryptography. DNA molecular computing uses a different computing paradigm and operates in analog rather than digital. This opens up a completely different phenomena that can be exploited to solve hard computation problems. Adleman demonstrated that an instance of the Hamiltonian Path Problem (HPP), an NP-complete problem, can be solved in polynomial time using a DNA biological computer [59]. For more information on DNA computing and cryptography, see [59][60][61][62][63][64][65][66][67].

9. References

References

[1] A. K. Lenstra and H. W. Lenstra, *The Development of the Number Field Sieve*. Berlin: Springer, 1993.

[2] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997. [Online]. Available: http://epubs.siam.org/doi/10.1137/S0097539795293172

[3] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, "Strengths and Weaknesses of Quantum Computing," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1510–1523, Oct. 1997. [Online]. Available: http://epubs.siam.org/doi/10.1137/S0097539796300933

[4] A. Montanaro, "Quantum Algorithms: An Overview," *npj Quantum Information*, vol. 2, no. 1, p. 15023, Nov. 2016, arXiv: 1511.04206. [Online]. Available: http://arxiv.org/abs/1511.04206

[5] L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, "Report on Post-Quantum Cryptography," National Institute of Standards and Technology, Tech. Rep. NIST IR 8105, Apr. 2016. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf

[6] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Available: http://portal.acm.org/citation.cfm?doid=359340.359342

[7] D. E. Knuth, *The Art of Computer Programming, Volume 3: (2nd Ed.) Sorting and Searching*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 1998.

[8] J. M. Pollard, "A Monte Carlo Method for Factorization," *BIT*, vol. 15, no. 3, pp. 331–334, Sep. 1975. [Online]. Available: http://link.springer.com/10.1007/BF01933667

[9] J. M. Pollard, "Theorems on Factorization and Primality Testing," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 76, no. 03, pp. 521–528, Nov. 1974. [Online]. Available: http://www.journals.cambridge.org/abstract_S0305004100049252

[10] H. W. Lenstra, "Factoring Integers with Elliptic Curves," *The Annals of Mathematics*, vol. 126, no. 3, p. 649, Nov. 1987. [Online]. Available: https://www.jstor.org/stable/1971363?origin=crossref

[11] M. A. Morrison and J. Brillhart, "A Method of Factoring and the Factorization of $F\sb{7}$," *Mathematics of Computation*, vol. 29, no. 129, pp. 183–205, 1975. [Online]. Available: http://www.ams.org/jourcgi/jour-getitem?pii=S0025-5718-1975-0371800-5

[12] C. Pomerance, "The Quadratic Sieve Factoring Algorithm," in *Advances in Cryptology*, T. Beth, N. Cot, and I. Ingemarsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, vol. 209, pp. 169–182. [Online]. Available: http://link.springer.com/10.1007/3-540-39757-4_17

[13] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann, "Factorization of a 768-Bit RSA Modulus," in *Advances in Cryptology – CRYPTO 2010*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and T. Rabin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 6223, pp. 333–350. [Online]. Available: http://link.springer.com/10.1007/978-3-642-14623-7_18

[14] P. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*. Santa Fe, NM, USA: IEEE Comput. Soc. Press, 1994, pp. 124–134. [Online]. Available: http://ieeexplore.ieee.org/document/365700/

[15] N. Xu, J. Zhu, D. Lu, X. Zhou, X. Peng, and J. Du, "Erratum: Quantum Factorization of 143 on a Dipolar-Coupling Nuclear Magnetic Resonance System [Phys. Rev. Lett. **108** , 130501 (2012)]," *Physical Review Letters*, vol. 109, no. 26, p. 269902, Dec. 2012. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.109.269902

[16] N. S. Dattani and N. Bryans, "Quantum Factorization of 56153 with only 4 Qubits," *arXiv:1411.6758 [quant-ph]*, Nov. 2014, arXiv: 1411.6758. [Online]. Available: http://arxiv.org/abs/1411.6758

[17] J. H. Silverman, *The Arithmetic of Elliptic Curves*, ser. Graduate Texts in Mathematics. New York, NY: Springer New York, 2009, vol. 106. [Online]. Available: http://link.springer.com/10.1007/978-0-387-09494-6

[18] L. Adleman, "A Subexponential Algorithm for the Discrete Logarithm Problem with Applications to Cryptography," in *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*. San Juan, Puerto Rico: IEEE, Oct. 1979, pp. 55–60. [Online]. Available: http://ieeexplore.ieee.org/document/4568001/

[19] V. S. Miller, "Use of Elliptic Curves in Cryptography," in *Advances in Cryptology — CRYPTO '85 Proceedings*, H. C. Williams, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986, vol. 218, pp. 417–426. [Online]. Available: http://link.springer.com/10.1007/3-540-39799-X_31

[20] N. Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–203, Jan. 1987. [Online]. Available: http://www.ams.org/jourcgi/jour-getitem?pii=S0025-5718-1987-0866109-5

[21] J. M. Pollard, "Monte Carlo Methods for Index Computation (mod p)," *Mathematics of Computation*, vol. 32, no. 143, p. 918, Jul. 1978. [Online]. Available: https://www.jstor.org/stable/2006496?origin=crossref

[22] S. Pohlig and M. Hellman, "An Improved Algorithm for Computing Logarithms over GF(p) and its Cryptographic Significance (Corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 1, pp. 106–110, Jan. 1978. [Online]. Available: http://ieeexplore.ieee.org/document/1055817/

[23] J. Proos and C. Zalka, "Shor's Discrete Logarithm Quantum Algorithm for Elliptic Curves," *arXiv:quant-ph/0301141*, pp. 317–344, Jan. 2003, arXiv: quant-ph/0301141. [Online]. Available: http://arxiv.org/abs/quant-ph/0301141

[24] P. Kaye, "Optimized Quantum Implementation of Elliptic Curve Arithmetic over Binary Fields," *Quantum Info. Comput.*, vol. 5, no. 6, pp. 474–491, Sep. 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=2011670.2011676

[25] S. Y. Yan, *Cybercryptography: Applicable Cryptography for Cyberspace Security*. Cham: Springer International Publishing, 2019. [Online]. Available: http://link.springer.com/10.1007/978-3-319-72536-9

[26] M. Mosca, "Cybersecurity in an Era with Quantum Computers: Will We Be Ready?" *IEEE Security & Privacy*, vol. 16, no. 5, pp. 38–41, Sep. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8490169/

[27] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, Y.-K. Liu, C. Miller, D. Moody, R. Peralta, R. Perlner, A. Robinson, and D. Smith-Tone, "Status report on the first round of the NIST post-quantum cryptography standardization process," National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. NIST IR 8240, Jan. 2019. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf

[28] A. K. Lenstra, H. W. Lenstra, and L. Lovsz, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, Dec. 1982. [Online]. Available: http://link.springer.com/10.1007/BF01457454

[29] H. Lenstra and P. Stevenhagen, "Book Review: Solving the Pell equation," *Bulletin of the American Mathematical Society*, vol. 52, no. 2, pp. 345–351, Dec. 2014. [Online]. Available: http://www.ams.org/bull/2015-52-02/S0273-0979-2014-01483-1/

[30] I. W. on Post Quantum Cryptography, *Post-quantum cryptography*, D. J. Bernstein, J. Buchmann, and E. Dahmen, Eds. Berlin Heidelberg: Springer, 2009, oCLC: 551314023.

[31] P. van Emde-Boas, *Another NP-complete partition problem and the complexity of computing short vectors in a lattice*, ser. Report. Department of Mathematics. University of Amsterdam. Department, Univ., 1981. [Online]. Available: https://books.google.co.uk/books?id=tCQiHQAACAAJ

[32] M. Ajtai, "The shortest vector problem in L_2 is NP -hard for randomized reductions (extended abstract)," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*. Dallas,

Texas, United States: ACM Press, 1998, pp. 10–19. [Online]. Available: http://portal.acm.org/citation.cfm?doid=276698.276705

[33] M. Ajtai, R. Kumar, and D. Sivakumar, "A sieve algorithm for the shortest lattice vector problem," in *Proceedings of the thirty-third annual ACM symposium on Theory of computing - STOC '01*. Hersonissos, Greece: ACM Press, 2001, pp. 601–610. [Online]. Available: http://portal.acm.org/citation.cfm?doid=380752.380857

[34] R. Kannan, "Improved algorithms for integer programming and related lattice problems," in *Proceedings of the fifteenth annual ACM symposium on Theory of computing - STOC '83*. Not Known: ACM Press, 1983, pp. 193–206. [Online]. Available: http://portal.acm.org/citation.cfm?doid=800061.808749

[35] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing - STOC '05*. Baltimore, MD, USA: ACM Press, 2005, p. 84. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1060590.1060603

[36] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *Journal of the ACM*, vol. 56, no. 6, pp. 1–40, Sep. 2009. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1568318.1568324

[37] B. Applebaum, D. Cash, C. Peikert, and A. Sahai, "Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems," in *Advances in Cryptology - CRYPTO 2009*, S. Halevi, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 5677, pp. 595–618. [Online]. Available: http://link.springer.com/10.1007/978-3-642-03356-8_35

[38] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96*. Philadelphia, Pennsylvania, United States: ACM Press, 1996, pp. 99–108. [Online]. Available: http://portal.acm.org/citation.cfm?doid=237814.237838

[39] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, ser. STOC '08. New York, NY, USA: ACM, 2008, pp. 197–206. [Online]. Available: http://doi.acm.org/10.1145/1374376.1374407

[40] R. Lindner and C. Peikert, "Better Key Sizes (and Attacks) for LWE-Based Encryption," in *Topics in Cryptology – CT-RSA 2011*, A. Kiayias, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6558, pp. 319–339. [Online]. Available: http://link.springer.com/10.1007/978-3-642-19074-2_21

[41] E. Orsini, J. van de Pol, and N. P. Smart, "Bootstrapping BGV Ciphertexts with a Wider Choice of $$p$$ and $$q$$," in *Public-Key Cryptography – PKC 2015*, J. Katz, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, vol. 9020, pp. 673–698. [Online]. Available: http://link.springer.com/10.1007/978-3-662-46447-2_30

[42] R. Hiromasa, M. Abe, and T. Okamoto, "Packing Messages and Optimizing Bootstrapping in GSW-FHE," in *Public-Key Cryptography – PKC 2015*, J. Katz, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, vol. 9020, pp. 699–715. [Online]. Available: http://link.springer.com/10.1007/978-3-662-46447-2_31

[43] V. Lyubashevsky and D. Wichs, "Simple Lattice Trapdoor Sampling from a Broad Class of Distributions," in *Public-Key Cryptography – PKC 2015*, J. Katz, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, vol. 9020, pp. 716–730. [Online]. Available: http://link.springer.com/10.1007/978-3-662-46447-2_32

[44] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem: Extended abstract," in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, ser. STOC '09. New York, NY, USA: ACM, 2009, pp. 333–342. [Online]. Available: http://doi.acm.org/10.1145/1536414.1536461

[45] O. Regev, "The Learning with Errors Problem (Invited Survey)," in *2010 IEEE 25th Annual Conference on Computational Complexity*. Cambridge, MA, USA: IEEE, Jun. 2010, pp. 191–204. [Online]. Available: http://ieeexplore.ieee.org/document/5497885/

[46] V. Lyubashevsky, C. Peikert, and O. Regev, "On Ideal Lattices and Learning with Errors over Rings," in *Advances in Cryptology – EUROCRYPT 2010*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and H. Gilbert, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, vol. 6110, pp. 1–23. [Online]. Available: http://link.springer.com/10.1007/978-3-642-13190-5_1

[47] D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa, "Efficient Public Key Encryption Based on Ideal Lattices," in *Advances in Cryptology – ASIACRYPT 2009*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and M. Matsui, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 5912, pp. 617–635. [Online]. Available: http://link.springer.com/10.1007/978-3-642-10366-7_36

[48] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring-based public key cryptosystem," in *Algorithmic Number Theory*, G. Goos, J. Hartmanis, J. van Leeuwen, and J. P. Buhler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, vol. 1423, pp. 267–288. [Online]. Available: http://link.springer.com/10.1007/BFb0054868

[49] V. Lyubashevsky, C. Peikert, and O. Regev, "A Toolkit for Ring-LWE Cryptography," in *Advances in Cryptology – EUROCRYPT 2013*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, T. Johansson, and P. Q. Nguyen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, vol. 7881, pp. 35–54. [Online]. Available: http://link.springer.com/10.1007/978-3-642-38348-9_3

[50] C. Peikert, O. Regev, and N. Stephens-Davidowitz, "Pseudorandomness of ring-LWE for any ring and modulus," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing - STOC 2017*. Montreal, Canada: ACM Press, 2017, pp. 461–473. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3055399.3055489

[51] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, "Relations among notions of security for public-key encryption schemes," in *Advances in Cryptology — CRYPTO '98*, G. Goos, J. Hartmanis, J. van Leeuwen, and H. Krawczyk, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, vol. 1462, pp. 26–45. [Online]. Available: http://link.springer.com/10.1007/BFb0055718

[52] J. ding, "New cryptographic constructions using generalized learning with errors problem," Cryptology ePrint Archive, Report 2012/387, 2012, https://eprint.iacr.org/2012/387.

[53] C. Peikert, "Lattice Cryptography for the Internet," in *Post-Quantum Cryptography*, M. Mosca, Ed. Cham: Springer International Publishing, 2014, vol. 8772, pp. 197–219. [Online]. Available: http://link.springer.com/10.1007/978-3-319-11659-4_12

[54] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, "Post-quantum key exchange - a new hope," Cryptology ePrint Archive, Report 2015/1092, 2015, https://eprint.iacr.org/2015/1092.

[55] V. Singh, "A practical key exchange for the internet using lattice cryptography," Cryptology ePrint Archive, Report 2015/138, 2015, https://eprint.iacr.org/2015/138.

[56] E. Barker, "Recommendation for Key Management Part 1: General," National Institute of Standards and Technology, Tech. Rep. NIST SP 800-57pt1r4, Jan. 2016. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf

[57] E. Alkim, R. Avanzi, J. Bos, L. Ducas, A. Piedra, T. Pöppelmann, P. Schwabe, and D. Stebila, "NewHope," Algorithm Specifications and Supporting Documentation, 2018.

[58] C. Heath, "Advanced Security and Digital Forensics - eSecurity," https://github.com/craig-heath/ASDF---eSecurity, 2019.

[59] L. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, no. 5187, pp. 1021–1024, Nov. 1994. [Online]. Available: http://www.sciencemag.org/cgi/doi/10.1126/science.7973651

[60] D. Boneh, C. Dunworth, R. J. Lipton, and J. Sgall, "On the computational power of DNA," *Discrete Applied Mathematics*, vol. 71, no. 1-3, pp. 79–94, Dec. 1996. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0166218X96000583

[61] D. Bray, "Protein molecules as computational elements in living cells," *Nature*, vol. 376, no. 6538, pp. 307–312, Jul. 1995. [Online]. Available: http://www.nature.com/articles/376307a0

[62] Gramß, M. Grob, M. Mitchell, T. Pellizzari, and T. Gram, *Non-Standard Computation.* Somerset: Wiley VCH, 1998, oCLC: 1024270047. [Online]. Available: http://public.eblib.com/choice/publicfullrecord.aspx?p=4956945

[63] M. Guo, M. S.-H. Ho, and W.-L. Chang, "Fast parallel molecular solution to the dominating-set problem on massively parallel bio-computing," *Parallel Computing*, vol. 30, no. 9-10, pp. 1109–1125, Sep. 2004. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0167819104000894

[64] R. Lipton, "DNA solution of hard computational problems," *Science*, vol. 268, no. 5210, pp. 542–545, Apr. 1995. [Online]. Available: http://www.sciencemag.org/cgi/doi/10.1126/science.7725098

[65] S. A. El-Seoud, R. F. Mohamed, and S. Ghoneimy, "DNA Computing: Challenges and Application," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 11, no. 2, p. 74, Apr. 2017. [Online]. Available: http://online-journals.org/index.php/i-jim/article/view/6564

[66] M. S. Muhammad, Z. Ibrahim, S. Ueda, O. Ono, and M. Khalid, "DNA Computing for Complex Scheduling Problem," in *Advances in Natural Computation*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, L. Wang, K. Chen, and Y. S. Ong, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, vol. 3611, pp. 1182–1191. [Online]. Available: http://link.springer.com/10.1007/11539117_159

[67] L. Kari, G. Gloor, and S. Yu, "Using DNA to solve the Bounded Post Correspondence Problem," *Theoretical Computer Science*, vol. 231, no. 2, pp. 193–203, Jan. 2000. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0304397599001000