A Substitution-Based Method for Data Hiding in DNA Sequences

Amany E. El-deeb[a], Ashraf B. El-Sisi[b] and Anas Youssef [c]
Department of Computer Science
Faculty of Computers and Information
Menoufia University
amanyeldeeb505@gmail.com[a], ashrafelsisi@ ci.menofia.edu.eg[b] and
anas.youssef@ci.menofia.edu.eg[c]

***Abstract*—To transmit data securely between different parties, a variety of security approaches have been proposed in the literature. Specifically, DNA based cryptography and steganography approaches were used to secure data transmission. In this paper, a substitution-based method for data hiding in DNA sequences is proposed. In the proposed data hiding method, data is encoded using a binary coding rule then the data is hidden into a DNA sequence. The proposed method provides an enhancement on a previously proposed DNA substitution method named Least Significant Base method. The proposed enhancement is based on a simple idea that, to the best of our knowledge, was not applied before. It was noticed that the DNA Amino acids can be organized into groups where each DNA codon in one of the groups can be used to encode two bits of the hidden message rather than only one bit as proposed by the Least Significant Base method. Like the Least Significant Base method, the proposed method is blind, preserves the DNA original biological structure in the fake DNA sequence and provides no expansion in the DNA sequence. The proposed method is evaluated using a public DNA sequences dataset named BALIBASE. The evaluation results showed that the proposed method achieved about 50% increase in the data hiding capacity when compared with the Least Significant Base method. Moreover, the results showed that the proposed method resulted in significant decrease in the cracking probability of the Least Significant Base method.**

Keywords*—*DNA Steganography; DNA Substitution Method; DNA Security.

I. INTRODUCTION

Storing and transmitting data securely can be achieved using either cryptography [1], steganography [2] or both techniques. Cryptography means converting information from its normal plain form into an encrypted form. On the other hand, steganography means hiding information by hiding a secret message into a plain message seen by an observer. DNA-steganography is an important branch for research and enhancements [3]. In this paper a DNA-steganography-based data hiding method is proposed. The proposed method provides an enhancement of Least Significant Base (LSBase) substitution method [4] to hide a message in the DNA sequence while still preserving the original biological DNA functionality with a blind feature. LSBase method substitutes the third nucleotide base of the DNA codon according to the value of only one hidden bit. The proposed method provides an enhancement on LSBase method by hiding either one or two message bits in the third nucleotide base according to some conditions of amino acids properties. To the best of our knowledge, this enhancement has been applied before and it provides more data hiding capacity and lower cracking probability than the LSBase method.

To evaluate the proposed method, a public DNA sequences dataset named BALIBASE is used [5]. This dataset provides various DNA sequences that are used in this work for data hiding and extraction. The evaluation results showed that the proposed method achieved about 50% increase in the data hiding capacity and significant decrease in the cracking probability. This achievement happens while still addressing a set of challenges in DNA-based steganography as listed in [3]. These challenges were already addressed by LSBase method [3] and still addressed by our proposed method. The challenges are as follows. Firstly, the DNA original biological structure is preserved so that the message sniffer cannot detect the hidden message from the fake DNA sequence by detecting a change in its original biological structure. Secondly, the blind feature of the proposed method is applied without the need to transmit the original DNA sequence for data extraction. Finally, no expansion in the used DNA sequences is applied.

The rest of the paper is organized as follows. A brief biological background is introduced in section II. Related work is presented in section III. The proposed method is described in section IV. Section V describes the datasets used in the evaluation experiments together with the experimental setup and the performance evaluation metrics. Experimental results are presented and discussed in Section VI. Finally, conclusions and future work directions are summarized in section VII.

II. BACKGROUND

In this section a brief biological background about DNA sequences, RNA sequences and amino acids is introduced. This background information is essential for clear understanding of other sections to follow.

A. DNA

Deoxyribonucleic Acid (DNA) is composed of two chains that coil around each other where each chain is called a nucleotide [6]. Each nucleotide is composed of sugar called deoxyribose, a phosphate group and nucleotide bases. These nucleotide bases are adenine (A), guanine (G), cytosine (C) and thymine (T). DNA is used as a molecular tool for exploring theories because of its properties. DNA bio-chemistry operations are used as basic operations in the IT field.

B. RNA

Ribonucleic Acid (RNA) strands are created using DNA strands in a transcription process as shown in figure1 [7]. In this process, DNA nucleotide bases are replaced with their corresponding RNA nucleotide bases. Let Ex(Base1, Base2) be an exchange function between two corresponding nucleotide bases, Base 1 and Base2. Base 1 and Base 2 are the same nucleotide bases in case the bases are G, A and C. An exception is found in thymine (T) case where RNA replaces it with uracil (U). While in case T is Base 1, it is replaced with uracil (U) in its corresponding RNA sequence. A translation process is then applied to create a polypeptide which is a linear sequence that consists of a large set of amino acids that form a protein molecule.

C. Amino Acids

Transfer RNA (tRNA) carries amino acids and reads the messenger RNA (mRNA) which includes three nucleotides called a codon [8]. Amino acids are linked in an order which instructs the protein structure and its function. Table 1 [4] shows the 64 amino acid codons, while table 2 shows these same 64 codons rearranged in groups to be suitable for the proposed work while keeping the DNA biological

functions [9]. The main purpose of this rearrangement is to allocate three different groups to be used in DNA steganography where one or two bits are used in data hiding or the whole amino acid codon is simply ignored.

Group1, G1, shown in Table 1 has four codons which are considered as exceptions because of their inapplicability for data hiding. The reason for this is that each of the four codons does not have a corresponding amino acid. Tryptophan (Trp), methionine (Met) and a Stop codon UGA are excepted because of having a single codon, while AUA codon is also neglected in our proposed method. The second group G2 includes part1, P1, and part2, P2. P1 has pyrimidine base which includes U and C while P2 has purine base which includes A and G. Last group is G3 which always includes corresponding amino acids with pyrimidine and purine bases together.

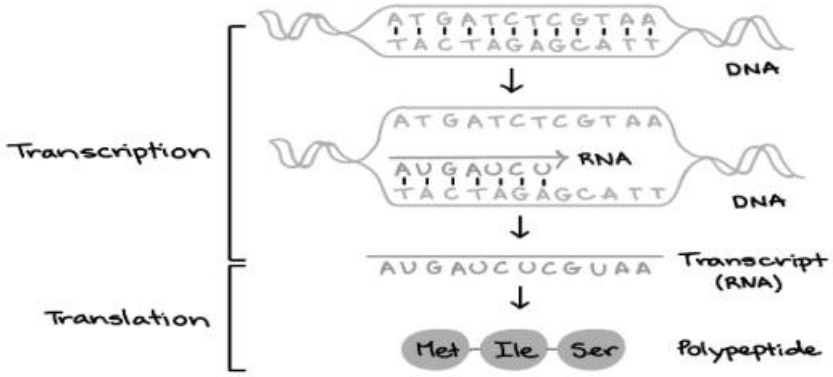

Figure 1: Transcription Process Using DNA Strands to Create RNA

III. RELATED WORK

Data hiding using DNA sequences are classified into three main methods as described in [10, 11]. These three methods are insertion-based, complementary-pair-based, and substitution-based. Each of the three methods is used to hide a secrete message in a reference DNA sequence at the sender's side then the hidden secrete message is extracted at the receiver's side. The focus of our work is on substitution-based method; however, we start by giving a brief description of sample research works that use the other two methods first, then we provide details about closely related substitution-based methods.

An insertion-based data hiding method in DNA sequences was proposed in [12]. It operates by hiding data which is encrypted by DNA and Amino Acids-Based Playfair cipher. In this work, the plaintext of the message is encoded into a DNA sequence. The DNA sequence is then encrypted using traditional Playfair encryption process. Both the encrypted message and the original DNA sequence are grouped into segments. Each encrypted message segment is inserted before a segment of the original DNA sequence to form the fake DNA sequence with the encrypted hidden message. This method sends an original DNA sequence to the receiver in addition to the fake DNA sequence that is used to hide the sent message. The original DNA sequence is used in the message extraction process. As opposed to the work proposed in this paper, this work can allow a message sniffer to extract the hidden message by comparing the fake DNA sequence with the original one if it was obtained.

Table 1: Amino Acids (64 codons) table Before Rearranging into Groups [4]

	U	C	A	G	
U	**Phe** UUU UUC **Leu** UUA UUG	**Ser** UCU UCC UCA UCG	**Tyr** UAU UAC **Stop** UAA UAG	**Cys** UGU UGC **Stop** UGA **Trp** UGG	**U**
C	**Leu** CUU CUC CUA CUG	**Pro** CCU CCC CCA CCG	**His** CAU CAC **Gln** CAA CAG	**Arg** CGU CGC CGA CGG	**C**
A	**Ile** AUU AUC AUA **Met/Start** AUG	**Thr** ACU ACC ACA ACG	**Asn** AAU AAC **Lys** AAA AAG	**Ser** AGU AGC **Arg** AGA AGG	**A**
G	**Yal** GUU GUC GUA GUG	**Ala** GCU GCC GCA GCG	**Asp** GAU GAC **Glu** GAA GAG	**Gly** GGU GGC GGA GGG	**G**

Table 2: Amino Acids Rearranged into Three Groups for the Proposed Work

G1	**Stop** UGA	**Ile** AUA	**Met/Start** AUG	**Trp** UGG
G2 P1	**Tyr** UAU UAC	**Cys** UGU UGC	**Phe** UUU UUC	**His** CAU CAC
	Asn AAU AAC	**Ser** AGU AGC	**Asp** GAU GAC	**Ile** AUU AUC
G2 P2	**Leu** UUA UUG	**Gln** CAA CAG	**Lys** AAA AAG	**Arg** AGA AGG
	Glu GAA GAG	**Stop** UAA UAG		
G3	**Pro** CCU CCC CCA CCG	**Leu** CUU CUC CUA CUG	**Arg** CGU CGC CGA CGG	**Gly** GGU GGC GGA GGG
	Yal GUU GUC GUA GUG	**Ala** GCU GCC GCA GCG	**Ser** UCU UCC UCA UCG	**Thr** ACU ACC ACA ACG

Another DNA-insertion-based method was proposed in [13]. This method uses a Binary XOR algorithm as follows. The method generates a random binary sequence, s, and the data message is split into a set of characters, M = {m1, m2, m3…mn}. The set M is then XORed with s to obtain another set A. The XORed binary sequence, A, is converted to the DNA protein sequence. Each of the set A and a reference DNA sequence S are divided into segments. Each segment of A is inserted before a segment of S using an insertion algorithm. As opposed to the work proposed in this paper, this method is unblind and it results in DNA sequence expansion since it is an insertion-based method.

A DNA-based data hiding scheme that uses complementary rules was proposed in [14]. In this work, a message is divided into segments which each segment contains two bits. The repeated characters in the reference DNA sequence are then selected. The message is hidden two by two bits only in the repeated characters of the DNA sequence. Each repeated character is exchanged by another character based on the agreed complementary rule between the sender and the receiver. This method had an efficient data hiding capacity with a low DNA sequence modification rate, and there is no expansion in the reference DNA sequence length. However, this scheme is not blind since it needs the original the DNA sequence for extraction which is one of challenges that are addressed in our proposed work.

Another DNA-based data hiding method that uses complementary rules was proposed in [15]. This method converts binary bits to DNA nucleotide bases by applying DNA base pairing rules. These nucleotide bases are then changed into other bases by applying complementary rules. An agreed public DNA sequence between the sender and the receiver is used to extract the index of each couple nucleotide bases numerically. The secret message can be hidden as numbers. Although, it is not a substitution method or a blind one, it helps in changing binary bit messages into DNA bases. This work explored DNA and its usage in steganography in addition to exploring binary coding scheme. As opposed to our work, a lake of the blind feature exists in this work because of the receiver's need to obtain the reference DNA sequence for extraction.

A third DNA-based data hiding method that uses a complementary rule was proposed in [16]. In this work, RSA algorithm was used in encrypting a message then the message is hidden using complementary character in the DNA sequence. As opposed to our work, the original DNA sequence is shared between both sender and receiver to be used in the extraction process.

A Table Lookup Substitution Method (TLSM) method is proposed in [17] as a DNA substitution-based method for data hiding. In this method with a rule table was used to replace the usage of complementary rules is. In essence, a 2-bit rule table was used to replace 1-bit complementary rule so that one nucleotide base can represent two bits of the hidden message. Moreover, two additional methods, a Base-t TLSM and an extended TLSM (ETLSM), were provided for more enhancements on the data hiding capacity. As opposed to our work, the message extraction process requires the existence of the original DNA sequence.

Another DNA-substitution based method for data hiding was proposed in [18]. In this work, A DNA-based Playfair cipher was applied for encryption first. Then, a DNA substitution method was used to hide the encrypted message in a DNA sequence. In this work, both cryptography and steganography methods were applied using DNA-based schemes which enhanced the security of the data hiding method. As opposed to our work, this work did not preserve the DNA sequence biological structure, provided an unblind scheme and resulted in expansion in the DNA sequence.

A third DNA-substitution based method for data hiding was proposed in [4]. Both DNA cryptography and steganography were implemented in this work. An encryption key is hidden by applying blind LSBase method, without affecting the DNA protein structure. The third nucleotide base of the codon is modified according to the value of the hidden message bit. Our proposed work provides enhancements to this method. Rather than only one message bit, one or two bits can be hidden in the third nucleotide base according to some conditions of amino acids properties. Our work is able to achieve about 50% higher data hiding capacity and much lower cracking probability than this work.

IV. PROPOSED METHOD

The proposed DNA steganographic method includes two algorithms, message hiding algorithm and the reverse message extraction algorithm. The two algorithms will be described in detail in the next subsections followed by a worked example.

A. Message Hiding Algorithm

The message hiding algorithm is listed in Algorithm 1. It takes an input M which is the secret message to be hidden in the original DNA sequence, S. The algorithm starts by converting S to RNA sequence, S_{RNA}, as shown in lines 2 to 4. S_{RNA} sequence is then grouped into codons in line 5. Each codon is denoted by $\mathbf{s_i}$ where i is a sequence number for the codon in S_{RNA}. In the loop that spans lines 6 to 24, the secret message bit(s), m_k where k is a sequence number for the group of secret message bit(s), are hidden using the codons generated from line 4 by only changing the third nucleotide base of the codon. This nucleotide base is referred to as the least significant base (LSBase) of the codon. The resulting fake RNA codon which holds the hidden message bit in its LSBase is denoted by Sm_{RNAi}. The groups used in the lines 6 to 24 of the message hiding algorithm are those listed before in Table 1. Specifically, the first group, G1, contains four ignored codons as discussed before. The second group, G2, includes part1 and part2 which hides only one bit. Part1 has pyrimidine base where U is used to represent 0 and C is used to represent 1. Part2 has purine base where A is used to represent 0 and G is used to represent 1. The last group, G3, is always used to represent two bits but if the remaining message is only one bit, another bit is created randomly which is deleted in the extraction process as will be shown later in Algorithm 2. The resulting fake SM_{RNA} is then converted back to its corresponding SM_{DNA} as shown in lines 25 to 27 to keep the biological characteristics of the resulting DNA sequence.

B. Message Extraction Algorithm

The message receiver applies the message extraction algorithm as listed in Algorithm 2. The message extraction process starts by converting SM_{DNA} sequence to SM_{RNA} as shown in lines 2 to 4. SM_{RNA} is then grouped into codons as shown in line 5. The LSBase of each codon is then used to extract the hidden bit as shown in lines 6 to 23. Specifically, if the codon belongs to G1, it is ignored, and the next codon is evaluated. If the codon belongs to G2 p1, the message is extracted as 0 or 1 if the LSBase of the codon is U or C, respectively. If the codon belongs to G2 p2, it is extracted as 0 or 1 if the LSBase of the codon is A or G, respectively. The message is extracted as two bits only if it is hidden in a codon that belongs to G3. Finally, as shown in line 24, a modification on the extracted message is made when the recovered message length is greater than the input message length, n. In such case the last bit is deleted. This is to compensate for the randomly added bit in line 17 of Algorithm 1.

Algorithm 1: Message hiding algorithm.

Input: M is the secret message in bits, S is the original DNA sequence
Output: SM_{DNA} is the fake DNA sequence which hides the message M

1. **Begin**
2. **For** each nucleotide base in S **do**
3. convert each T to U to obtain S_{RNA}
4. **End For**
5. Group the $\mathbf{S_{RNA}}$ into codons
6. **For** each codon s_i in $\mathbf{S_{RNA}}$ **do** until M is fully hidden
7. **If** ($\mathbf{s_i} \in \mathbf{G_1}$) **Then** ignore s_i **End If**
8. **If** ($\mathbf{s_i} \in \mathbf{G_2\ P_1}$) **Then**
9. **If** ($\mathbf{m_k} = 0$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by U **End If**
10. **If** ($\mathbf{m_k} = 1$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by C **End If**
11. **End If**
12. **If** ($\mathbf{s_i} \in \mathbf{G_2\ P_2}$) **Then**
13. **If** ($\mathbf{m_k} = 0$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by A **End If**
14. **If** ($\mathbf{m_k} = 1$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by G **End If**
15. **End If**
16. **If** ($\mathbf{s_i} \in \mathbf{G_3}$) **Then**
17. **If** ($\mathbf{m_k}$ is only one bit) **Then** add another bit randomly **End If**
18. **If** ($\mathbf{m_k} = 00$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by A **End If**
19. **If** ($\mathbf{m_k} = 01$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by C **End If**
20. **If** ($\mathbf{m_k} = 10$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by G **End If**
21. **If** ($\mathbf{m_k} = 11$) **Then** LSBase of $\mathbf{sm_{RNAi}}$ is substituted by U **End If**
22. **End If**
23. $SM_{RNA} = SM_{RNA} + sm_{RNAi}$
24. **End For**
25. **For** each nucleotide base in SM_{RNA} **do**
26. convert each U to T to obtain SM_{DNA}
27. **End For**
28. **End**

C. Worked Example

Table 3 and Table 4 show the outputs obtained after applying Algorithm 1 and Algorithm 2, respectively, on a worked example. Suppose that M is 100111110 and S is TAAAACCCAAAATCTTGACTG. Algorithm 1 will be performed as shown in the following steps.

1. Convert S_{DNA} sequence to S_{RNA} to obtain S_{RNA}: UAAAACCCAAAAUCUUGACUG
2. Group S_{RNA} into codons to obtain S_{RNA} = UAA|AAC|CCA|AAA|UCU|UGA|CUG
3. Apply the proposed substitution method to obtain the data shown in Table 3. The result SM_{RNA} will be UAG|AAU|CCC|AAG|UCU|UGA|CUG.
4. Convert SM_{RNA} back to SM_{DNA} to obtain SM_{DNA} = TAGAATCCCAAGTCTTGACTG

Algorithm 2 is then performed as shown in the following steps.

1. Convert fake SM_{DNA} to SM_{RNA} sequence to obtain SM_{RNA} = UAGAAUCCCAAGUCUUGACUG
2. Group SM_{RNA} into codons: SM_{RNA} = UAG|AAU|CCC|AAG|UCU|UGA|CUG
3. Extract the message, M, from SM_{RNA} to obtain M = 100111110.

Algorithm 2: Message extraction algorithm

Input: SM_{DNA} is the fake DNA sequence that carries the secret message M, n is the message length
Output: M is the recovered secret message in bits
1. **Begin**
2. **For** each nucleotide base in SM_{DNA} **do**
3. convert each T to U to obtain SM_{RNA}
4. **End For**
5. Group the SM_{RNA} into codons
6. **For** each received codon $\mathbf{s_i}$ in SM_{RNA} **do** until M is fully extracted
7. **If** ($s_I \in G_1$) **Then** ignore s_i **End If**
8. **If** ($s_i \in G_2\ p_1$) **Then**
9. **If** (LSBase of sm_{RNAi} is U) **Then** $m_k = 0$ **End If**
10. **If** (LSBase of sm_{RNAi} is C) **Then** $m_k = 1$ **End If**
11. **End If**
12. **If** ($s_i \in G_2\ p_2$) **Then**
13. **If** (LSBase of sm_{RNAi} is A) **Then** $m_k = 0$ **End If**
14. **If** (LSBase of sm_{RNAi} is G) **Then** $m_k = 1$ **End If**
15. **End If**
16. **If** ($s_i \in G_3$) **Then**
17. **If** (LSBase of sm_{RNAi} is A) **Then** $m_k = 00$ **End If**
18. **If** (LSBase of sm_{RNAi} is C) **Then** $m_k = 01$ **End If**
19. **If** (LSBase of sm_{RNAi} is G) **Then** $m_k = 10$ **End If**
20. **If** (LSBase of sm_{RNAi} is U) **Then** $m_k = 11$ **End If**
21. **End If**
22. $M = M + m_k$
23. **End For**
24. **If** (M's actual message length > n) **Then** delete the last bit **End If**
25. **End**

Table 3: Outputs obtained after applying Algorithm 1 on a worked example

S Codon	UAA	AAC	CCA	AAA	UCU	UGA	CUG
Amino acid group	G2p2	G2p1	G3	G2p2	G3	G1	G3
Message bit	1	0	01	1	11	ignore	10
Fake SM Codon	UAG	AAU	CCC	AAG	UCU	UGA	CUG

Table 4: Outputs obtained after applying Algorithm 2 on a worked example

Fake SM Codon	UAG	AAU	CCC	AAG	UCU	UGA	CUG
Amino acid group	G2p2	G2p1	G3	G2p2	G3	G1	G3
Message bit	1	0	01	1	11	ignore	10

V. EXPERIMENTAL SETUP AND EVALUATION METRICS

This section describes the datasets used in the performance evaluation experiments together with the experimental setup and the performance evaluation metrics. BAliBASE version 1.0 [5] is a database of manually refined multiple sequence alignments which is a way of arranging the sequences of DNA, RNA or protein. BAliBASE version 2.0 [5] is an extended version of BAliBASE version 1.0 with extra three DNA reference datasets. BAliBASE version 2.0 contains eight reference datasets that hold 167 DNA sequences. BAliBASE version 2.0 is used in this work. Tables 5 and 6 show a brief description of four different reference datasets that are part of the BAliBASE version 2.0 database [5]. These datasets are used in the evaluation of the proposed work.

Table 5: Reference Datasets Specifications -1

1pamA-ref-set2 dataset			2myr-ref-set2 dataset		
DNA Sequence Name	**Number of nucleotide bases**	**Number of codons**	**DNA Sequence Name**	**Number of nucleotide bases**	**Number of codons**
1pamA	1470	490	**2myr**	1422	474
amy_thetu	1470	490	**bgls_trirp**	1446	482
amy_bacci	1473	491	**bglc_maize**	1446	482
cdgt_bacli	1467	489	**bgl2_bacsu**	1398	466
cdg2_bacma	1470	490	**lacg_staau**	1386	462
cdg1_bacma	1473	491	**lacg_lacac**	1395	465
cdgt_bacst	1467	489	**lacg_lacca**	1389	463
cdgt_bacs2	1458	486	**bgla_bacpo**	1320	440
amym_bacst	1482	494	**bglb_bacpo**	1323	441
cdgt_klepn	1533	511	**bgla_thema**	1317	439
amyb_bacpo	1347	449	**bgla_bacci**	1317	439
amy1_schpo	1395	465	**bgla_clotm**	1329	443
ydd2_schpo	1374	458	**bgls_agrsp**	1320	440
2aaa	1407	469	**bgls_calsa**	1347	449
amya_aspor	1404	468	**lph_human**	1389	463
amy1_schoc	1401	467	**lph_rabit**	1398	466
amy1_sacfi	1377	459	**1gowA**	1389	463
1jdc	1305	435			

As shown in Tables 5 and 6, there are four datasets named: 1pamA-ref-set2**,** 2myr-ref-set2, 1lvl-ref-set2 and 1wit-ref-set2. For every reference dataset, the name and the number of nucleotide bases that form every DNA sequence used in the experiments are shown. For each DNA sequence, the number of nucleotide bases is divided by three to compute the total number of codons that form the DNA sequence. The secret message bits can consume all or part of these codons depending on the message length. The last reference dataset named 1wit-ref-set2 which is shown in Table 6 differs from the other three reference datasets in that its average number of nucleotide bases is much lower than the other three reference datasets. The reference dataset, 1wit-ref-set2, is used to evaluate the case where the secret message length is larger than the DNA sequence length. In such case the secret message bits are hidden using more than one DNA sequence from the same reference dataset.

The proposed message hiding, and extraction algorithms are implemented in Java. In this work every reference DNA sequence is used to hide the same secret message. The secret message used in the evaluation experiments is "**DNA Computing.** " statement which contains 15 ASCII characters for a total of 120 bits in binary form as 01000100 01001110 01000001 00100000 01100011 01101111 01101101 01110000 01110101 01110100 01101001 01101110 01100111 00101110 00100000.

Two performance evaluation metrics are used compare our proposed method with previous works. The performance metrics are the data hiding capacity [4,15] and cracking probability [3]. The time complexity of our proposed method is also presented but with no comparisons with other related methods which did not specify their time complexities. The performance metrics are described as follows.

Table 6: Reference Datasets Specifications -2

1lvl-ref-set2 dataset			**1wit-ref-set2 dataset**		
DNA Sequence Name	**Number of nucleotide bases**	**Number of codons**	**DNA Sequence Name**	**Number of nucleotide bases**	**Number of codons**
1lvl	1309	436.3	**1wit**	279	93
dldh_halvo	1387	462.3	**cpsf_chick**	279	93
dld1_bacst	1378	459.3	**cpss_human**	282	94
dldh_trybb	1396	465.3	**myms_human**	285	95
dldh_human	1396	465.3	**vca1_mouse**	255	85
dldh_alceu	1411	470.3	**1tlk**	291	97
dldh_ecoli	1393	464.3	**cavt_brala**	291	97
dldh_yeast	1419	473.0	**dcc_human**	288	96
dld2_bacsu	1411	470.3	**ptpf_human**	300	100
dldh_zymmo	1378	459.3	**mpsf_human**	285	95
dldh_mycge	1357	452.3	**pgbm_mouse**	264	88
mera_staau	1384	461.3	**iml2_drome**	267	89
mera_strli	1399	466.3	**vgr2_mouse**	279	93
mera_pseae	1366	455.3	**nca2_xenla**	276	92
tytr_crifa	1408	469.3	**axo1_chick**	273	91
3grs_1	1348	449.3	**nrg_drome**	288	96
gshr_pseae	1333	444.3	**caml_rat**	261	87
gshr_anasp	1360	453.3	**cont_mouse**	276	92
gshc_arath	1357	452.3	**lar_drome**	285	95
gshr_burce	1321	440.3	**1hnf**	318	106
gshr_caeel	1387	462.3			
gshr_arath	1372	457.3			
1nhp	1278	426.0			

A. Data Hiding Capacity

The data hiding capacity shows the number of bits that can be hidden using a given original DNA sequence. In this work, three data hiding capacity metrics are used namely, Maximum Data Hiding Capacity (MDHC) and Actual Data hiding Capacity (ADHC) [4] and Bit Per Nucleotide base (BPN) [4,10,12]. As described previously, the DNA codons used for message hiding can either be used for data hiding, i.e., *consumed* for secret message substitution, or can be ignored. The MDHC represents the maximum number of message bits that can be hidden using all codons of each DNA sequence. This is

calculated by using all codons in the DNA sequence without ignoring any of them. Each codon that belongs to groups G1 and G2 can be used to hide one message bit, while each codon that belongs to group G3 can be used to hide two message bits.

The ADHC represents the actual number of message bits that can be hidden using a given original DNA sequence. This is calculated by using all codons in the DNA sequence but this time with ignoring the codons that belong to group G1 while using each codon that belongs to group G2 and group G3 to hide one message bit and two message bits, respectively. The percentage of the ADHC to the MDHC is denoted by the Consumed data hiding Capacity Percentage (CCP).

Finally, the BPN represents the average number of bits that can be hidden using each nucleotide base. This metric is calculated by dividing the length of the secret message in bits by the total number of consumed nucleotide bases. The number of consumed nucleotide bases is calculated by multiplying the total number of consumed codons by three.

B. Cracking Probability

The cracking probability is defined as the probability of attacker's success in revealing the secret message for a specific algorithm [3]. Suppose a reference DNA sequence, *S*, is used to hide a secret message, *M*, the cracking probability can be computed for any message hiding algorithm by the following equation:

$$CP_{general\_algorithm} = \frac{1}{F1} * \frac{1}{F2} * \ldots * \frac{1}{F9} \qquad (1)$$

where the factors *F1* to *F9* are summarized as follows and more details can be found in [3]. Note that including as many valid factors as possible in equation (1) leads to lower cracking probability. This depends on the method used for hiding data in DNA sequences.

- ***F1*: Size of *S* and *M* is required.** If the secret message extraction process requires the size of *S* and *M*, and it is assumed that *n* is the length of the faked DNA sequence, then there will be *(n-1)* possibilities, i.e. guesses, required to obtain the size of *S* and *M*.
- ***F2*: Number of segments that form *M***. When *M* is divided into *n* segments and *n* must be known for message extraction, then there will be 2^{m-1} possibilities required to obtain the number of segments that form M, where *m* is the length of M.
- ***F3*: Number of segments that form *S*.** If *S* is divided into *p* segments which must be known for extraction, then there are 2^{s-1} possibilities required to obtain the number of segments that form *S*, where s is the length of *S*.
- ***F4*: Binary coding rule**. A binary coding rule is required if *M* is mapped from its original format to a binary coding format. For example, suppose that nucleotide base A is mapped to one of the following four binary forms 00, 01, 10 or 11. If nucleotide base A is mapped to 00 then nucleotide base C will have three possibilities of binary forms which are 01, 10 or 11 and so on. Therefore, the total number of possibilities for the binary coding rule is 4!.
- ***F5*: DNA reference sequence is selected from a public database.** The public databases contain millions of real DNA sequences. If the algorithm works by selecting a DNA sequence from one of these public databases, then the possibilities required to guess one of these DNA sequences is equal to the total number of such DNA sequences. For example, GenBank database [19] has 212 million DNA

sequence and EBI public database [20] has 163 million DNA sequences. As stated in [3] the total number of possibilities to guess one of the DNA sequences is 163 million if the EBI public database is used in the evaluation.

- ***F6*: Table lookup rule is used in the substitution method**. If a lookup table is used to assign the substitution rules on which the algorithm depends, then the number of guesses required to discover the used substitution rules is equal to the total number of entries stored in the lookup table.
- ***F7*: Conversion function is used in the substitution method**. If a conversion function is used to hide *M* by substituting some of *S* nucleotide bases, then *M* cannot be extracted without knowing this conversion function.
- ***F8*: Complementary rules are used**. If *M* is hidden using complementary rules, such rules should be discovered for the extraction of *M*. Therefore, the number of guesses required to discover the used complementary rules is equal to the total number of rules.
- ***F9*: Injective mapping rule is used in the substitution method.** In this rule, each two bits in *M* are hidden using one DNA nucleotide base according to the value of the two bits. Therefore, the number of guesses required to discover the used injective mapping rules is equal to the total number of rules.

We note that not all the above factors are applicable when our proposed work is used. The only factors that are applied to our work are *F1*, *F4*, *F5*, *F7* and *F9*, however, other parameters are not included due to the following reasons. Firstly, *F2* and *F3* are used only to evaluate the insertion-based methods due to segmentation of the message and the DNA sequence. They are not used with substitution-based methods like the proposed one. Secondly, *F6* is used in case a look up table is applied in the substitution method, and it is not applied in our proposed work. Finally, *F8* is used when complementary rules are used in the message hiding and extraction which were not used in the proposed work.

C. Time Complexity

The time complexity provides a theoretical measure of the time required to run each of the developed message hiding and extraction algorithms. This measure is provided in terms of the input size which is mainly dependent on the message length in bits.

VI. EXPERIMENTAL RESULTS

In this section the results of evaluating the proposed message hiding and extraction algorithms are presented and discussed.

A. Data Hiding Capacity of the Proposed Approach

The experimental results obtained using the BAliBASE 1pamA-ref-set2 DNA reference dataset are shown in Table 7. The results listed in Table 7 are described as follows. For each DNA sequence listed in Table 7, both the number of codons used for substitution and the number of ignored codons are listed. For example, the secret message that is used in the experiments whose length is 120 bits is hidden using 83 codons of 1pamA DNA sequence while 8 codons are ignored from this sequence. The codons used for substitution can either be used to hide 1 bit or 2 bits. For example, 1pamA DNA sequence provides 46 and 37 codons to hide 1 bit and 2 bits of the secret message, respectively. The percentage of the number of *ignored* codons to the *total consumed* codons, i.e., used and ignored codons together, is computed for each DNA sequence. For example, 1pamA DNA sequence dedicates 91 codons to hide the secret

message, while 8 codons are ignored, this results in a percentage of the number of ignored codons to the total consumed codons 8.97%.

Table 7: Data Hiding Capacity of the Proposed Substitution Method Using **1pamA-ref-set2** reference dataset

DNA sequence name	Used codons	Hide 1 bit	Hide 2 bits	Ignored codons	Ignored/total consumed (%)	MDHC (bits)	ADHC (bits)	CCP (%)	BPN
1pamA	83	46	37	8	8.79	128	120	94	0.44
amy_thetu	80	40	40	4	4.76	124	120	97	0.48
amy_bacci	82	44	38	9	9.89	129	120	93	0.44
cdgt_bacli	85	50	35	2	2.29	122	120	98	0.46
cdg2_bacma	81	41	40	8	8.98	129	121	94	0.45
cdg1_bacma	81	42	39	3	3.57	123	120	98	0.48
cdgt_bacst	82	44	38	7	7.86	127	120	94	0.45
cdgt_bacs2	85	50	35	9	9.57	129	120	93	0.43
amym_bacst	80	39	41	7	8.04	128	121	95	0.46
cdgt_klepn	81	42	39	4	4.70	124	120	97	0.47
amyb_bacpo	89	58	31	7	7.29	127	120	94	0.42
amy1_schpo	86	52	34	11	11.34	131	120	92	0.41
ydd2_schpo	84	48	36	6	6.66	126	120	95	0.44
2aaa	84	48	36	10	10.63	130	120	92	0.43
amya_aspor	81	42	39	7	7.95	127	120	94	0.45
amy1_schoc	83	45	38	7	7.77	128	121	95	0.45
amy1_sacfi	86	52	34	6	6.52	126	120	95	0.43
1jdc	82	43	39	8	8.88	129	121	94	0.45

The MDHC and ADHC of the proposed data hiding process are shown in bits for each DNA sequence listed in Table 7. For example, 1pamA DNA sequence dedicates 91 codons to hide the secret message. This results in a MDHC of 128 bits. The ADHC is either 120 bits or 121 bits if an extra bit is added when necessary as shown previously in the message hiding algorithm that was described in Algorithm 1. The CCP is about 94%. The average CCP across all DNA sequences listed in Table 7 is around 95%.

Finally, the BPN is calculated for each DNA sequence listed in Table 7. For example, when 91 codons from 1pamA DNA sequence are used to hide the secret message, this results in a total of 273 nucleotide bases. Therefore, to hide the 120 bits of the secret message, the BPN is about 0.44. The average BPN across all DNA sequences listed in Table 7 is about 0.45.

Tables 8, 9 and 10 show similar results for the other three BAliBASE reference data sets used in the experiments. Table 11 shows a summary of the results shown in Tables 7, 8, 9 and 10 for all reference datasets used in the evaluation experiments.

B. Cracking Probability of the Proposed Approach

In this section, the proposed message hiding algorithm is analyzed based on the above described cracking probability parameters to get the number of possibilities required by the attacker to extract the secret message. The proposed message hiding algorithm is analyzed based on the parameters *F1*, *F4*, *F5*, *F7* and *F9* as follows. Other parameters are not applicable as described previously in Section V-B.

- *F1=n−1* where *n* is the length of the faked DNA sequence as described above.
- *F4=4! =24*. This is because *M* may be extracted as two bits only if it is hidden in a codon that belongs to G3 as shown previously in Algorithm 2. Therefore, if nucleotide base A is mapped to 00 then nucleotide base C will have three possibilities of binary forms which are 01, 10 or 11 and so on. Therefore, the total number of possibilities for the binary coding rule is 4! = 4*3*2*1 = 24.
- *F5= 167*. This is because the number of guesses to obtain the selected DNA reference sequence that hides the message is equal to the number of DNA sequences located in BaliBASE version 2.0 that is used in the evaluation of this work.
- *F7=2! ∗2! =4.* The reason for this is that proposed data hiding algorithm substitutes the pyrimidine nucleotide bases, i.e., T and C, with pyrimidine nucleotide bases and substitutes the purine nucleotide bases, i.e., A and G, with purine nucleotide bases. This happens for codons that belong to G2 as shown previously in Algorithm 1. Since nucleotide base T is represented by either 0 or 1 and nucleotide base C is represented by either 0 or 1 so the number of different probabilities is 2!. Similarly, for nucleotide bases A and G. Therefore, the total number of guesses required to obtain this conversion function is 2! ∗2! =4.
- *F9 = 2!*. The reason for this is that the number of injective mappings that can be established between the possible nucleotide base substitutions and the possible binary coding rules of two secret bits is 4! = 24. There are 4 binary coding rules in G3 as shown previously in Algorithm 2. This group is used to hide 2 bits with four different possibilities and 4 nucleotide bases A, C, G, U.

Table 8: Data Hiding Capacity of the Proposed Substitution Method Using **2myr-ref-set2** reference dataset

DNA sequence same	Used codons	Hide 1 bit	Hide 2 bits	Ignored codons	Ignored/total consumed (%)	MDHC (bits)	ADHC (bits)	CCP (%)	BPN
2myr	81	42	39	5	5.81	125	120	96	0.47
bgls_trirp	82	43	39	6	6.81	127	121	95	0.46
bglc_maize	85	50	35	8	8.60	128	120	94	0.43
bgl2_bacsu	82	43	39	8	8.88	129	121	94	0.45
lacg_staau	80	40	40	4	4.76	124	120	97	0.48
lacg_lacac	83	45	38	6	6.74	127	121	95	0.45
lacg_lacca	81	42	39	3	3.57	123	120	98	0.48
bgla_bacpo	82	44	38	9	9.89	129	120	93	0.44
bglb_bacpo	80	39	41	9	10.11	130	121	93	0.45
bgla_thema	81	42	39	9	10.00	129	120	93	0.44
bgla_bacci	82	44	38	4	4.65	124	120	97	0.47
bgla_clotm	87	54	33	9	9.37	129	120	93	0.42
bgls_agrsp	83	46	37	7	7.77	127	120	94	0.44
bgls_calsa	84	48	36	6	6.66	126	120	95	0.44
lph_human	83	46	37	6	6.74	126	120	95	0.45
lph_rabit	80	39	41	6	6.97	127	121	95	0.47
1gowA	82	44	38	9	9.89	129	120	93	0.44

Therefore, the proposed message hiding algorithm's cracking probability can be computed by:

$$CP_{proposed\_algorithm} = \frac{1}{(n-1)} * \frac{1}{24} * \frac{1}{167} * \frac{1}{2!*2!} * \frac{1}{4!} \qquad (2)$$

Table 9: Data Hiding Capacity of the Proposed Substitution Method Using **1lvl-ref-set2** reference dataset

DNA sequence name	Used codons	Hide 1 bit	Hide 2 bits	Ignored codons	Ignored/total Consumed (%)	MDHC (bits)	ADHC (bits)	CCP (%)	BPN
1lvl	75	30	45	5	6.25	125	120	96	0.50
dldh_halvo	77	33	44	7	8.33	128	121	95	0.48
dld1_bacst	75	30	45	4	5.06	124	120	97	0.51
dldh_trybb	75	29	46	3	3.84	124	121	98	0.52
dldh_human	79	38	41	4	4.81	124	120	97	0.48
dldh_alceu	80	39	41	1	1.23	122	121	99	0.50
dldh_ecoli	76	32	44	3	3.79	123	120	98	0.51
dldh_yeast	80	39	41	1	1.23	122	121	99	0.50
dld2_bacsu	74	28	46	2	2.63	122	120	98	0.53
dldh_zymmo	77	33	44	4	4.93	125	121	97	0.50
dldh_mycge	84	48	36	3	3.44	123	120	98	0.46
mera_staau	78	36	42	3	3.70	123	120	98	0.49
mera_strli	71	21	50	1	1.38	122	121	99	0.56
mera_pseae	75	29	46	3	3.84	124	121	98	0.52
tytr_crifa	77	33	44	5	6.09	126	121	96	0.49
3grs_1	77	33	44	4	4.93	125	121	97	0.50
gshr_pseae	75	30	45	1	1.31	121	120	99	0.53
gshr_anasp	78	35	43	3	3.70	124	121	98	0.50
gshc_arath	75	30	45	1	1.31	121	120	99	0.53
gshr_burce	76	31	45	3	3.79	124	121	98	0.51
gshr_caeel	83	46	37	4	4.59	124	120	97	0.46
gshr_arath	80	40	40	1	1.23	121	120	99	0.49
1nhp	82	43	39	8	8.88	129	121	94	0.45

C. Comparing the Proposed Method with Related Works

To compare the proposed method with related work, the same BAliBASE reference datasets are used to evaluate the related approaches. Table 12 shows the comparison between the proposed method with related works in terms of five different comparison criteria. The comparison criteria are the data hiding capacity, if the DNA biological structure is persevered, if blindness feature is applicable, if DNA sequence expansion occurs and the cracking probability. Firstly, the data hiding capacity is measured in BPN as described previously. Secondly, the DNA biological structure preservation means that the fake DNA sequence which contains the hidden secret message can still preserve the biological structure of the original DNA sequence. If the fake DNA sequence preserves the DNA biological structure of the original DNA sequence, this increases the security level of the message hiding algorithm because the modification

made in the fake DNA sequence cannot be detected by the message sniffer by visual inspection. Thirdly, The application of blindness feature means that the original DNA sequence is not sent with the fake DNA sequence for extraction of the hidden secret message. This also increases the security level of the message hiding algorithm because even if the message sniffer could be able to obtain the fake DNA sequence, the message sniffer cannot be able to extract the hidden secret message since he/she does know the original DNA sequence. Fourthly, the Expansion that may occur in the DNA sequence means that there is an increase of the DNA sequence size after holding the hidden message. Finally, the cracking probability as previously defined is computed for each method used in the comparison.

Table 10: Data Hiding Capacity of the Proposed Substitution Method Using **1wit-ref-set2** reference dataset

DNA sequence Name	Used codons	Hide 1 bit	Hide 2 bits	Ignored codons	Ignored/total consumed (%)	MDHC (bits)	ADHC (bits)	CCP (%)	BPN
1wit	81	42	39	4	4.70	124	120	97	0.47
cpsf_chick	79	37	42	3	3.65	124	121	98	0.49
cpss_human	81	42	39	6	6.89	126	120	95	0.46
myms_human	79	38	41	2	2.46	122	120	98	0.49
vca1_mouse	79	42	37	6	7.05	122	116	95	0.45
1tlk	87	49	38	5	5.43	130	125	96	0.45
cavt_brala	84	48	36	4	4.54	124	120	97	0.45
dcc_human	80	40	40	6	6.97	126	120	95	0.47
ptpf_human	81	41	40	3	3.57	124	121	98	0.48
mpsf_human	84	48	36	4	4.54	124	120	97	0.45
pgbm_mouse	73	26	47	6	7.59	126	120	95	0.51
iml2_drome	82	44	38	6	6.81	126	120	95	0.45
vgr2_mouse	78	36	42	6	7.14	126	120	95	0.48
nca2_xenla	86	51	35	6	6.52	127	121	95	0.44
axo1_chick	77	34	43	4	4.93	124	120	97	0.49
nrg_drome	79	38	41	2	2.46	122	120	98	0.49
caml_rat	81	41	40	4	4.70	125	121	97	0.47
cont_mouse	82	43	39	7	7.86	128	121	95	0.45
lar_drome	78	35	43	3	3.70	124	121	98	0.50
1hnf	92	64	28	6	6.12	126	120	95	0.41

Table 11: Summary of all reference datasets used in experiments

Dataset	1pamA-ref-set2	2myr-ref-set2	1lvl-ref-set2	1wit-ref-set2
Average Ignored/total consumed (%)	7.53	7.49	3.93	5.39
Average CCP (%)	94.65	94.74	97.42	96.29
Average BPN	0.45	0.45	0.50	0.47

The comparison results listed in Table 12 show the following observations. The proposed work in [10] applies three different methods namely: insertion, complimentary rules and substitution methods. The best of three methods in terms of data hiding capacity is the substitution method. This method provides a higher data hiding capacity, 0.82, than our proposed approach; however, it does not preserve the DNA

biological structure and does not apply blindness like our proposed work. This also applies to the complementary rules proposed in [16]. On the contrary, the LSBase method proposed in [4] provides a lower data hiding capacity than our proposed approach, 0.333, but it can still preserve the DNA biological structure and applies blindness. An efficient data hiding capacity is obtained with no expansion in our proposed approach where it can hide one bit and two bits of message. This data hiding capacity is not the best capacity in this comparison, but it is still efficient. Like the work proposed in [4], our proposed approach still provides two strong points in conserving DNA biological structure and blindness. The DNA sequence expansion occurs in [10] and [13] since they are insertion-based methods.

Recall that using more factors in equation (1) of the cracking probability implies better data hiding method in terms of this metric. Among all compared methods, the cracking probability is the best in both insertion-based method [10] because of utilizing many segmentation factors and in the insertion-based method [18] due to using XOR operation. The worst method in terms of cracking probability is in the substitution-based method [10]. Our proposed method achieved significant decrease in the cracking probability than the LSBase substitution-based method [4].

Table 12: Comparing the proposed method with related works

Approach	Implementation Method	BPN	DNA Biological Structure is preserved?	Blindness Applicable?	Expansion occurs in DNA Sequence?	Cracking probability
Shiu et al., 2010 [10]	Insertion-based method	0.58	no	no	yes	$\frac{1}{(n-1)} * \frac{1}{24} * \frac{1}{167} * \frac{1}{2^{s-1}} * \frac{1}{2^{m-1}}$
	Complementary rules	0.07	no	no	no	$\frac{1}{24} * \frac{1}{167}$
	Substitution-based method	0.82	no	no	no	$\frac{1}{6} * \frac{1}{167}$
Mitras and Abo, 2013 [16]	Complementary rules	0.80	no	no	no	$\frac{1}{24} * \frac{1}{167} * \frac{1}{24}$
Malathi et al., 2017 [13]	Insertion-based method	1.5	no	no	yes	$\frac{1}{(n-1)} * \frac{1}{24} * \frac{1}{167} * \frac{1}{2^{s-1}} * \frac{1}{2^{m-1}} * \frac{1}{2^{8m}}$
Khalifa, 2013 [4]	LSBase	0.333	yes	yes	no	$\frac{1}{24} * \frac{1}{167} * \frac{1}{2!*2!}$
Our Proposed Method	Substitution-based method	0.45 - 0.5	yes	yes	no	$\frac{1}{(n-1)} * \frac{1}{24} * \frac{1}{167} * \frac{1}{2!*2!} * \frac{1}{4!}$

D. Time Complexity of the Proposed Algorithms

In this section the time complexity of the proposed message hiding, and extraction algorithms are presented. The time complexity both algorithms are the same. Suppose that the secret message length is *n* bits, the time complexity of the each of the message hiding and message extraction algorithms is linear with respect to *n* as shown in the following equation.

$$T(n) = O(n) \qquad (3)$$

This is because the message hiding algorithm shown previously in Algorithm 1 and the message extraction algorithm shown previously in Algorithm 2 work on the secret message bit by bit until the whole message is fully hidden or fully extracted, respectively.

VII. CONCLUSIONS AND FUTURE WORK

In this work an enhanced substitution-based method for data hiding using DNA sequences is proposed. The proposed method adds an enhancement on a previous work [4] by hiding either one or two message bits in an original DNA sequence. This results in about 50% increase in the data hiding capacity and significant decrease in the cracking probability when compared with the work in [4]. Like the work in [4], the proposed method is blind since the original DNA sequence is not transferred between transmission parties. Moreover, the proposed approach preserves the DNA original biological structure in the fake DNA sequence that is transmitted to the receiver. There is no expansion that happens in the fake DNA sequence that is transmitted to the receiver.

The proposed message hiding, and message extraction algorithms are presented and discussed in this work. Both algorithms are implemented in Java and are evaluated using a public DNA BAliBASE database. Three different evaluation metrics were used in the evaluation experiments. These are the data hiding capacity, cracking probability, and the time complexity. The results showed that the proposed message hiding algorithm achieved more efficient data hiding capacity than some of the related work while still persevering DNA original biological structure and blindness.

Future work includes improving the data hiding capacity of the proposed work in addition to enhancing the security of the proposed work by using more factors to lower the cracking probability. This can be done by adding more features to the proposed approach that can make other not applicable cracking probability factors to be applicable in our work.

REFERENCES

1. A. Cherian, SR. Raj and A. Abraham, “A Survey on different DNA cryptographic methods,” *International Journal of Science and Research*, vol. 2, no. 4, April, pp. 167-169, 2013.
2. CT. Clelland, . Risca and C. Bancroft, “Hiding messages in DNA microdots,” *Nature*, vol. 399, no. 6736, pp. 533-534, 1999.
3. G. Hamed, M. Marey, S. El-Sayed, and F. Tolba. "DNA based steganography: survey and analysis for parameters optimization." In *Applications of intelligent optimization in biology and medicine*, Springer, Cham, 2016, pp. 47-89.
4. A. Khalifa, “LSBase: A key encapsulation scheme to improve hybrid crypto systems using DNA steganography,” In Proc. IEEE International Conference on Computer Engineering & Systems’08, 2013, pp. 105-110.
5. J. D. Thompson, F. Plewniak and O. Poch, “BAliBASE: a benchmark alignment database for the evaluation of multiple alignment programs,” *Bioinformatics (Oxford, England)*, vol. 15, no. 1, pp. 87-88, 1999.
6. S. A. El-Seoud, R. Mohamed and S. Ghoneimy, “DNA Computing: Challenges and Application,” *International Journal of Interactive Mobile Technologies*, vol. 11, no. 2, pp. 74-87, 2017.
7. M. Borda and O. Tornea, “DNA secret writing techniques,” In Proc. IEEE International Conference on Communications ’08, 2010, pp. 451-456.
8. F.E. Ibrahim, M.I. Moussa and H.M. Abdalkader, “A symmetric encryption algorithm based on DNA computing,” *International Journal of Computer Applications*, vol. 97, no. 16, pp. 41-45, 2014.

9. M. Sabry, M. Hashem, T. Nazmy and ME. Khalifa, “A DNA and amino acids-based implementation of playfair cipher,” *International Journal of Computer Science and Information Security*, vol. 8, no. 3, pp.129-136, 2010.
10. H. J. Shiu, K. L. Ng, J.F. Fang, R.C. Lee and C. H. Huang, “Data hiding methods based upon DNA sequences,” *Information Sciences*, vol. 180, no. 11, pp. 2196-2208, 2010.
11. O. A. Al-Harbi, W.E. Alahmadi and A.O. Aljahdali, “Security analysis of DNA based steganography techniques,” *SN Applied Sciences*, vol. 2, no. 2, pp. 1-10, 2020.
12. A. Atito, A. Khalifa and S.Z. Rida, “DNA-based data encryption and hiding using playfair and insertion techniques,” *Journal of Communications and Computer Engineering*, vol. 2, no. 2, pp. 44-49, 2012.
13. P. Malathi, M. Manoaj, R. Manoj, V. Raghavan and R.E. Vinodhini, “Highly improved DNA based steganography,” *Procedia Computer Science*, vol. 115, pp. 651 -659, 2017.
14. C. Guo, C. C. Chang and Z. H. Wang, “A new data hiding scheme based on DNA sequence,” *international Journal of Innovative Computing, Information and Control*, vol. 8, no. 1, pp. 139-149, 2012.
15. M.R. Abbasy, P. Nikfard, A. Ordi and M.R. Torkaman, “DNA base data hiding algorithm,” *International Journal of New Computer Architectures and their Applications*, vol. 2, no. 1, pp.183-192, 2012.
16. B. A. Mitras and A.K. Abo, “Proposed steganography approach using DNA properties,” *International journal of information technology and business management*, vol 14, no. 1, pp. 96-102, 2013.
17. J. S. Taur, H.Y. Lin, H. L. Lee and C.W. Tao, “Data hiding in DNA sequences based on table lookup substitution,” *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 10, pp. 6585-6598, 2012.
18. A. Khalifa and A. Atito, “High-capacity DNA-based steganography,” In Proc. IEEE International Conference on Informatics and Systems ’08, 2012, pp. BIO-76-BIO-80.
19. D. Benson, I. Karsch-Mizrachi, D. Lipman, J. Ostell and D. Wheeler. “GenBank,” *Nucleic acids research*, vol. 41, no. D1, pp. D36-D42, 2012.
20. P. Rocca-Serra, A. Brazma, H. Parkinson, U. Sarkans, M. Shojatalab, S. Contrino, J. Vilo, N. Mukherjee, G. Abeygunawardena, E. Holloway, M. Kapushesky and others, “ArrayExpress: a public database of gene expression data at EBI,” *Comptes rendus biologies*, vol. 326, no. 10-11, pp. 1075-1078, 2003.