Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/13S118

An Algorithm for Arithmetic Operations: An Application in SInsDelP System

V. Sudha[1,*] *and K. S. Easwarakumar*[2]

[1] Department of Computer Science & Engineering, Kumaraguru College of Technology, Coimbatore 641 049, India.
[2] Department of Computer Science & Engineering, Anna University, Chennai 600 025, India.

Received: 3 Jun. 2019, Revised: 2 Jul. 2019, Accepted: 23 Jul. 2019
Published online: 1 Aug. 2019

Abstract: In this paper, a novel algorithm for performing the basic arithmetic operations on different number systems such as decimal, octal and hexadecimal is proposed. To facilitate this algorithm, a new number representation is implemented. This representation accepts the input numbers as such without converting to intermediate forms. This proposed procedure can be extended for any number system.

Keywords: DNA computation, replicative transposition, arithmetic operations, decimal numbers, octal numbers, hexadecimal numbers, number representation

1 Introduction

Computer has become a part of the human life. Hence, the computation that we do in a computer should be very fast. Most of the operations that are performed on a computer are realized using the basic arithmetic operations such as addition and subtraction. Many algorithms are proposed to design fast adder and subtractor.

Computing is the process of performing operations with the help of a machine called a computer. There are different types of computing theorems such as cloud computing [1,2,3,4], quantum computing [5,6,7,8] and DNA computing [9,10]. DNA computing is a molecular computing that replaces silicon in the computer by DNA. DNA computing makes DNA as a computational medium. A success of a computing model depends on the implementation of the basic operations. DNA computing, a kind of molecular computing, is one of the fast-growing areas of research. Research in DNA computing was initiated by Adleman in 1994 by solving an instance of Hamiltonian Path Problem (HPP) [11].

There have been many methodologies proposed for performing arithmetic operations using DNA computing. A NAND operation is implemented in [19] by generating all possible strands with respect to the truth table. Complementary strands anneal to represent the output. In [40], string substitution is used for performing basic boolean operations. A recursive procedures are used for implementing the basic arithmetic operations [14]. Each integer is represented as a binary number. Here, DNA encoding is done for representing each integer. In [13] and [49], tile self-assembly is used wherein all possible values for each arithmetic operation are represented as tiles. Stem loops are used in [47] for representing the four possible combination of the binary digits 0's and 1's. In multiplication, the output digit is 1 only if both input digits are 1. Thus, only in this case the stem loops expand to form linear double chain structure, while in all other cases the structure of the stem loops are retained. In currently-proposed algorithms, a decimal number given as input is translated into binary number and the operations are applied on it. Hence, a new methodology is proposed for performing arithmetic operations on any number system using the operations defined in *SInsDelP* system [36]. This methodology performs the operations directly on the given numbers without converting to binary equivalent.

In DNA computing, a problem can be solved either practically using the biological operations in laboratories or it is solved manually and verified theoretically with the help of theorems. There are various insertion and deletion theoretical models proposed in the literature.

* Corresponding author e-mail: sudhavelvizhi@gmail.com

In [37], it is shown that using only either insertion or deletion operation or both, universal turing machine can be derived. In [33], a context-sensitive insertion-deletion model (*INSDEL*) is proposed. Theoretical model for inserting circular DNA strand into a linear DNA strand is proposed in [16]. An insertion-deletion system based on replicative transposition and PCR-site specific mutagenesis (*SInsDelP*) is proposed in [36]. In this paper, an algorithm for performing arithmetic operations on different number system is proposed.

The present paper is organized as follows. Section 2 reviews the operations defined in *SInsDelP* insertion-deletion model. In Section 3, a new method of number representation is proposed. Algorithms for decimal operations are proposed in Section 4. Furthermore, Section 5 gives the operations necessary for octal arithmetic operations. Finally, the operations necessary for hexadecimal arithmetic is explained in Section 5.

2 SInsDelP System

Definition 1. *A SInsDelP [35] system is a quadruple*

$$ID = (V, T, A, P)$$

where V is an alphabet, $T \subseteq V$ is the terminal alphabet of ID, P=INS$\bigcup$DEL such that $INS \subseteq V^\star \times V^\star$ is the set of single contextual insertion rules, $DEL \subseteq V^\star \times V^\star \times V^\star$ is the set of contextual deletion rules and $A \subseteq V^\star$ is the set of axioms. The deletion rules have higher precedence than the insertion rules when both can be applied simultaneously.

The rules in INS and DEL take the form $(x,y)_I$ and $(x,y,z)_D$ respectively. Here, x and z represent the context and y represents the content. The insertion operation in this system is implemented using replicative transposition. In replicative transposition, the context is replicated on either side of the inserted content. For illustration, let uxv be the string, then after insertion it becomes $uxyxv$ where $u, v \in V^{\star\star}$. Likewise, the deletion operation in *SInsDelP* system is realised using PCR-site specific mutagenesis. Here, the string that lies between the contexts is deleted. For instance, the string xyz becomes xz after deletion.

3 Number Representation

To solve a problem in DNA computing, first the problem must be encoded as DNA sequences. Here, the input for the algorithm is assumed to be a set of numbers of any base value. The following logic is proposed for encoding the numbers.

It is known that a number with base value n is a collection of digits ranging from 0 to n. To convert a given number to some equivalent value in DNA bases, two methods can be followed. One is following a representation that considers the given decimal number as it is. The other way is, to denote each digit in a number by DNA bases, whose concatenation gives the equivalent number value.

In this paper, the latter method is used for the representation. To facilitate this, a separator is used between the digits. Along with representing the digits in a number, the place value of the digits also to be remembered to facilitate the respective arithmetic operations. Thus, the following encoding schemes are proposed for representing a number. The base G^i is used to representing the face value of a digit i, i.e. to represent the digit value k, k instances of the base G will be used. The base T is used as the separator between digits and A^i denotes the positional value of the digit as the digit is i^{th} least significant digit.

To smooth the operation, two types of representations are used. One for representing the first number in the list, while the other for representing the latter numbers. For instance, if we want to add two numbers, $(315)_{10}$ and $(423)_{10}$, then the first number $(315)_{10}$ is encoded in one form, and the other number $(423)_{10}$ in another form. The encoding of the first number is done as described above. For example, the number $(315)_{10}$ is encoded as

$$(315)_{10} : A^4TG^3A^3TG^1A^2TG^5A^1T$$

In the above encoding, the number of G denotes the digit value. The A^4 present in most significant place is used only for algorithmic convenience.

To represent other numbers, the bases complementary to the bases in the above scheme can be used. The base C to represent the face value of a digit, the base A as the separator between digits and T^i denotes the positional value of the digit as the digit is i^{th} least significant one. For instance, the number $(315)_{10}$ is encoded as

$$(315)_{10} : T^4AC^3T^3AC^1T^2AC^5T^1A$$

The most significant digit T^4 is used only for algorithmic convenience. Note that the notation used in the above scheme are complimentary to each other which facilitates the bonding process. Also, it is presumed that initially, the test tube contains the DNA strand corresponding to the first number, while the strands of the remaining numbers are inserted one by one into the test tube.

The proposed number representation can be extended further for any number system. To this number representation, operations defined in *SInsDelP* system is applied to perform the necessary arithmetic operations. In this paper, algorithm for performing arithmetic operations on various number systems such as decimal, octal and hexadecimal is proposed.

4 Decimal Arithmetic Operations

In this section, the algorithms for performing arithmetic operations on decimal numbers are disscused

4.1 Decimal Addition

Here, the operations necessary for performing addition operation on given decimal numbers are defined. Also, an example for illustrating its operations is explained.

To facilitate further addition operation in future, it is assumed that the notation used for representing the first number is used for representing the final answer.

4.1.1 Logic Description

The following logic is used for performing addition operation. Initially, the DNA strand of the first number in the set of numbers to be added is inserted in to the test tube. Followed by it, the DNA strands of the consecutive numbers are added. When a DNA strands are further added, the newly added strand is appended at the end of the first DNA strand.

It is known that, when a set of numbers are to be added, then all the digits in a particular place value of the given numbers has to be added. To facilitate this, in the proposed system digits present in a particular place value are grouped together using percolation operations. This will move the DNA subsequence of the particular digit until the position value of the digit matches with the number of instances of the DNA base A. After percolation, the unnecessary DNA bases are removed from the resultant DNA strand.

Now, the number of instances of the bases G and C used for representing the digit value is summed and the result is represented in terms of G. After performing addition operation, there may be a possibility of existence of carry. In decimal arithmetic operations, a carry is said to occur when the number of $G^{\prime s}$ present is greater than 10 where 10 is the base value of the given number. This carry/extra G present is moved to higher level.

Sometimes, we may get a digit in the last position which is not bounded by A. Hence, an extra A is added at the left most end if such a carry exists. The above process is repeated until all the numbers in the input are added.

4.1.2 Procedure

The operations described in previous section are expressed using the insertion and deletion operations defined in the *SInsDelP* system. Operations for performing decimal addition in DNA computing is listed below.

1.$(AT, TA, B)_D$;
2.$(A^iTG^\star, T^iA, C^\star A)_D$;
3.$(A^iTG^\star, A^iTG^\star T^jAC^k, T^jAC^kA^iTG^\star)_D$;
4.$(T^jAC^kA^iTG^\star, A^iTG^\star T^jAC^k, T^{i-1}AC^\star)_D$;
5.$(A^kT, G^iC^j, G^{i+j})_D$;
6.$(G^{i+j}, G^iC^j, A^kT,)_D$;
7.$(A^{i+1}TG^\star, A^iTG^{10}, GA^iT)_D$;
8.$(B, A^iTG^{10}, GA^iT)_D$;
9.$(GA^iTG^\star, A^iTG^{10}, A^{i-1}TG^\star)_D$;
10.$(B, BGA^iT, BA^{i+1}GA^iT)_D$;
11.$(BA^{i+1}GA^iT, BGA^iT, G)_D$;
12.$(G, AT\lambda, AT\text{Second DNA Strand}\lambda)_D$;
13.$(AT\text{Second DNA Strand}\lambda, AT\lambda, \lambda)_D$;
14.$(AT\lambda, AT\text{Second DNA Strand}\lambda)_I$;
15.$(BGA^iT, BA^{i+1}GA^iT)_I$;
16.$(A^iTG^\star T^jAC^k, T^jAC^kA^iTG^\star)_I$; if $i > j$
17.$(G^iC^j, G^{i+j})_I$;
18.$(A^iTG^{10}, GA^iT)_I$.

When context for both insertion and deletion exists simultaneously, then the deletion operation takes higher priority.

4.1.3 Case Study

Let 724 and 456 be the two decimal numbers to be added. These two numbers are encoded as shown below.

$$724 : A^4TG^7A^3TG^2A^2TG^4AT$$

$$456 : T^4AC^4T^3AC^5T^2AC^6TA$$

First, the strand

$$A^4TG^7A^3TG^2A^2TG^4AT$$

is inserted into the test tube. In the above DNA strand, there is no context for any of the above operations and hence no operations is performed on it. Now, after inserting the second strand, the following sequence will happen. When a second DNA strand is inserted, it will be placed at the end of the first DNA strand as shown below using the operations indicated at the side of the DNA strand. By rules 12, 13, and 14 we obtain

$$A^4TG^7A^3TG^2A^2TG^4AT\boldsymbol{T^4AC^4T^3AC^5T^2AC^6TA}$$

Next, the embedded strand has to be percolated to its respective position as explained in section 4.1.1. By operations 16, 3, and 4 we obtains

$$A^4TG^7A^3TG^2A^2TG^4\boldsymbol{T^4AC^4AT}T^3AC^5T^2AC^6TA$$

Repeating the operations 16, 3, and 4 leads to the following DNA strand. As indicated here, the digits that belongs to same place value are grouped together at the end of the operation.

$$A^4TG^7\boldsymbol{T^4AC^4}A^3TG^2\boldsymbol{T^3AC^5}A^2TG^4\boldsymbol{T^2AC^6}ATTA.$$

Though the digits to be added are in the same position, still some extra symbols are in between them. These symbols are deleted using operations 1 and 2. The modified strand is given below. By operations 1 and 2 we obtains

$$A^4TG^7C^4A^3TG^2C^5A^2TG^4C^6AT$$

Now, the digits are added to get the required result. As stated, the resultant values are indicated in terms of base G to facilitate further operations. The DNA strand resulting from addition operation contains the duplication of context as given as follows:

$$A^4T\boldsymbol{G^7C^4}G^{11}\boldsymbol{G^7C^4}A^3T\boldsymbol{G^2C^5G^7G^2C^5}A^2T\boldsymbol{G^4C^6}G^{10}\boldsymbol{G^4C^6}AT.$$

Deleting these extra symbols yields the following strand and using operations 5 and 6 we obtain

$$A^4T\boldsymbol{G}^{11}A^3T\boldsymbol{G^7}A^2T\boldsymbol{G}^{10}AT.$$

In a decimal number, a digit can take values ranging from 0 to 9. Hence, when the result value exceeds 9, a carry is said to occur. This carry has to be propagated/summed up with the value in the next higher position. It is carried out by operations 18, 8 and 9, we obtain

$$\boldsymbol{GA^4TG}A^3TG^7\boldsymbol{GA^2T}AT.$$

After the carry propagation, the resulting strand begins with G. But according to the proposed representation it must begin with some A^i. This is carried out by operations 10, 11 and 15.

$$\boldsymbol{A^5TG}A^4TGA^3TG^8A^2TG^0AT$$

The above process is repeated until all the decimal numbers are processed.

The performance of the above process can further be improved by partitioning the set of numbers into subsets and applying the algorithm in parallel on these subsets. The sum of the results of these subset gives the final result.

4.2 Decimal Subtraction

Decimal subtraction is performed by subtracting subtrahend from minuend. The logic used for subtraction is same as that of the addition operation. The procedure used for performing subtraction operation along with an example is described in the following sections.

4.2.1 Logic description

Though the representation and logic used for subtraction is same as that of addition, the results obtained from subtraction operation can be positive or negative. When the output is positive, the final answer does not need any modification but when it is negative, the representation must be changed indicating that the result is negative. Here, the DNA base value C is used for indicating the negative result.

4.2.2 Procedure

Decimal subtraction in DNA computing using *SInsDelP* system is achieved using the following operations,

1.$(AT,TA,B)_D$;
2.$(A^iTG^\star,T^iA,C^\star A)_D$;
3.$(A^iTG^\star,A^iTG^\star T^jAC^k,T^jAC^kA^iTG^\star)_D$;
4.$(T^jAC^kA^iTG^\star,A^iTG^\star T^jAC^k,T^{i-1}AC^\star)_D$;
5.$(A^kT,G^iC^j,G^{i-j})_D$;
6.$(G^{i-j},G^iC^j,A^kT)_D$;
7.$(A^kT,G^iC^j,C^{i-j})_D$;
8.$(C^{i-j},G^iC^j,A^kT)_D$;
9.$(A^kT,G^iA^+TC^k,G^{i-1}A^+TG^{10-k})_D$;
10.$(G^{i-1}A^+TG^{10-k},G^iA^+TC^k,A^kT)_D$;
11.$(B,A^{i+1}T,A^iT)_D$;
12.$(G,AT\lambda,AT$ Second DNA Strand$\lambda)_D$;
13.$(AT$ Second DNA Strand$\lambda,AT\lambda,\lambda)_D$;
14.$(AT\lambda,AT$ Second DNA Strand$\lambda)_I$;
15.$(A^iTG^\star T^jAC^k,T^jAC^kA^iTG^\star)_I$; if $i>j$
16.$(G^iC^j,G^{i-j})_I$; if $i\geq j$
17.$(G^iC^j,C^{j-i})_I$; if $i<j$
18.$(G^iA^+TC^k,G^{i-1}A^+TG^{10-k})_I$;

These operations are specified according to the priorities. Low-priority operation takes place only if context for high-priority operation does not exists in the DNA strand. Operations are listed in the order of their priority. It is illustrated by the following example.

4.2.3 Case Study 1

Let 123 be subtracted from 452. These two numbers are encoded as shown below.

$$452: A^4TG^4A^3TG^5A^2TG^2AT$$

$$123: T^4AC^1T^3AC^2T^2AC^3TA$$

As per our assumption, first

$$A^4TG^4A^3TG^5A^2TG^2AT$$

is inserted into the test tube. In the above DNA strand there is no context for any of the above operations and hence it waits for further strands to be inserted into the tube. Next, the DNA strand of the second number is inserted and gets embedded behind the first strand as shown below, by operations 12, 13, and 14 we obtain

$$A^4TG^4A^3TG^5A^2TG^2AT\boldsymbol{T^4AC^1T^3AC^2T^2AC^3TA}$$

After insertion as context for strand percolation exists, it gets percolated until the respective position is reached. This performed by the operation 15.

$$A^4TG^4A^3TG^5A^2TG^2\boldsymbol{ATT^4AC^1T^4AC^1ATATT^4AC^1}$$

$$T^3AC^2T^2AC^3TA.$$

The additional symbols that gets generated due to the above insertion operation is deleted using the instructions 3 and 4.

$$A^4TG^4A^3TG^5A^2TG^2\boldsymbol{T^4AC^1AT}T^3AC^2T^2AC^3TA$$

Repeating operations 15, 3, and 4 results in the following DNA strand,

$$A^4TG^4\boldsymbol{T^4AC^1}A^3TG^5\boldsymbol{T^3AC^2}A^2TG^2\boldsymbol{T^2AC^3}ATTA$$

Now, the extra symbols between the two operands are deleted using operations 1 and 2. As decided, the final output DNA strand follows the representation of the first DNA strand

$$A^4TG^4C^1A^3TG^5C^2A^2TG^2C^3AT$$

To the above DNA strand, subtraction operation is applied using the operations 16 and 17. After deleting extra symbols, it is observed that the digit in ones place of first number(4) is smaller than the digit in ones place of second number(6). Hence, a value must be borrowed from the adjacent digit. Base C is used for this purpose. By operations 5, 6, 7 and 8, we obtain

$$A^4T\boldsymbol{G^3}A^3T\boldsymbol{G^3}A^2T\boldsymbol{C^1}AT$$

The following logic is used for performing the borrow operation in the proposed algorithm. Since the input here is a decimal number (base 10), the borrow operation is performed by subtracting the corresponding digit from 10. Thus, subtracting 1 from 10 yields 9. This is shown by the following DNA strand. By operations 18, 9 and 10, we obtain

$$A^4T\boldsymbol{G^3}A^3T\boldsymbol{G^2}A^2T\boldsymbol{G^9}AT$$

The above process is repeated until all the decimal numbers to be subtracted are processed.

4.2.4 Case Study 2

The example in this subsection illustrates the flow of the algorithm when some of the digits in the given input are equal.

Let 41 be subtracted from 45. Using the above proposed scheme, these two numbers will be encoded as shown below.

$$45 : A^3TG^4A^2TG^5AT$$

$$41 : T^3AC^4T^2AC^1TA$$

As per our assumption first

$$A^3TG^4A^2TG^5AT$$

is inserted into the test tube followed by the DNA strand of the second number. Now, after inserting the second strand, the operations 12, 13 and 14 takes place. After applying these operations, the resulting DNA strand looks like the strand given below.

$$A^3TG^4A^2TG^5AT\boldsymbol{T^3AC^4T^2AC^1TA}$$

Followed by this embedding operation, percolation of second DNA strand takes place as shown below. By operation 15, we obtain

$$A^3TG^4A^2TG^5\boldsymbol{ATT^3AC^4T^3AC^4ATATT^3AC^4}T^2AC^1$$

TA Deleting the extra symbols, by operations 3, and 4 we obtain

$$A^3TG^4A^2TG^5\boldsymbol{T^3AC^4AT}T^2AC^1TA$$

Recursively applying the above described operations yields,

$$A^3TG^4\boldsymbol{T^3AC^4}A^2TG^5\boldsymbol{T^2AC^1}ATTA$$

Now, deleting the separator symbols used in the representation of second DNA strand using operations 1 and 2 leads to the following string. By operations 1 and 2, we obtain

$$A^3TG^4C^4A^2TG^5C^1AT$$

All the preprocessing necessary for the subtraction operation is done. Next, the subtraction operation is performed by applying the operations 16, 17, 5, 6, 7 and 8 as given below. By operations 16 and 17, we obtain

$$A^3T\boldsymbol{G^4C^4G^0G^4C^4}A^2T\boldsymbol{G^5C^1}G^4\boldsymbol{G^5C^1}AT$$

By operations 5, 6, and 7, we obtain

$$A^3T\boldsymbol{G^0}A^2T\boldsymbol{G^4}AT$$

In this example, as the value of the most significant bit is same in both the inputs, the last digit is 0 in the result. By operation 11, we obtain

$$A^2T\boldsymbol{G^4}AT$$

4.2.5 Case Study 3

In subtraction operation, when minuend is lesser than the subtrahend then the result obtained will be negative. In the proposed algorithm, the presence of the base C at the most significant bit indicates the negative result. This is illustrated in detail by the example in this section.

Let 62 be subtracted from 59. These two numbers are encoded as shown below.

$$59: A^3TG^5A^2TG^9AT$$

$$62: T^3AC^6T^2AC^2TA$$

As per our assumption first

$$A^3TG^5A^2TG^9AT$$

is inserted into the test tube which is followed by second DNA strand. After inserting the second strand, the following sequence takes place. By operations 12, 13, and 14, we obtain

$$A^3TG^5A^2TG^9AT\boldsymbol{T^3AC^6T^2AC^2TA}.$$

By operation 15, we obtain

$$A^3TG^5A^2TG^9\boldsymbol{ATT^3AC^6T^3AC^6ATATT^3AC^6}T^2AC^2TA.$$

According to operations 3, and 4, we obtain

$$A^3TG^5A^2TG^9\boldsymbol{T^3AC^6AT}T^2AC^2TA$$

Now, applying operations 13, 3, and 4 recursively leads to,

$$A^3TG^5\boldsymbol{T^3AC^6}A^2TG^9\boldsymbol{T^2AC^2}ATTA$$

Based on operations 1 and 2

$$A^3TG^5C^6A^2TG^9C^2AT.$$

By operations 16 and 17

$$A^3T\boldsymbol{G^5C^6C^1G^5C^6}A^2T\boldsymbol{G^9C^2G^7G^9C^2}AT.$$

From operations 5, 6, and 7, we obtain

$$A^3T\boldsymbol{C^1}A^2T\boldsymbol{G^7}AT$$

In the above strand, there is no context for operation 8. Hence, the iteration stops with this step. Presence of C in the left most end of the final strand indicates that the result is a negative number.

At this stage, either the subtraction operation can be repeated by interchanging subtrahend and minuend or any restoring operation can be applied to get the actual result. For instance, in the above example the number 17 can be subtracted from 20 to get the actual result 13.

5 Octal Arithmetic Operations

The logical difference that lies between the decimal and octal arithmetic operations is in the range of values taken by each number system. In octal number system, as the base value is 8, the range of value taken by a digit in an octal number is 0 to 7.

Algorithm for performing arithmetic operations on octal numbers is described in the following subsection.

5.1 Addition Operation

The operations necessary for addition operation are enumerated below.

1.$(AT, TA, B)_D$;
2.$(A^iTG^\star, T^iA, C^\star A)_D$;
3.$(A^iTG^\star, A^iTG^\star T^jAC^k, T^jAC^kA^iTG^\star)_D$;
4.$(T^jAC^kA^iTG^\star, A^iTG^\star T^jAC^k, T^{i-1}AC^\star)_D$;
5.$(A^kT, G^iC^j, G^{i+j})_D$;
6.$(G^{i+j}, G^iC^j, A^kT,)_D$;
7.$(A^{i+1}TG^\star, A^iTG^8, GA^iT)_D$;
8.$(B, A^iTG^8, GA^iT)_D$;
9.$(GA^iTG^\star, A^iTG^8, A^{i-1}TG^\star)_D$;
10.$(B, BGA^iT, BA^{i+1}GA^iT)_D$;
11.$(BA^{i+1}GA^iT, BGA^iT, G)_D$;
12.$(G, AT\lambda, AT\text{Second DNA Strand}\lambda)_D$;
13.$(AT\text{Second DNA Strand}\lambda, AT\lambda, \lambda)_D$;
14.$(AT\lambda, AT\text{Second DNA Strand}\lambda)_I$;
15.$(BGA^iT, BA^{i+1}GA^iT)_I$;
16.$(A^iTG^\star T^jAC^k, T^jAC^kA^iTG^\star)_I$; if $i > j$
17.$(G^iC^j, G^{i+j})_I$;
18.$(A^iTG^8, GA^iT)_I$;

5.1.1 Case Study

Let 324 and 256 be the two octal numbers to be added. Encoding of these two numbers using DNA bases are given below.

$$324: A^4TG^3A^3TG^2A^2TG^4AT$$

$$256: T^4AC^2T^3AC^5T^2AC^6TA$$

Initially, the strand

$$A^4TG^3A^3TG^2A^2TG^4AT$$

is inserted into the test tube. Followed by it, the DNA strand corresponding to the next numbers are inserted into the test tube resulting in the following operations. By rules 12, 13, and 14, we obtain

$$A^4TG^3A^3TG^2A^2TG^4AT\boldsymbol{T^4AC^2T^3AC^5T^2AC^6TA}$$

Next, the digits in the second strand are percolated to the respective digits(digits in particular position) as explained in the previous section. From by operations 16, 3, and 4, we obtain

$$A^4TG^3A^3TG^2A^2TG^4\boldsymbol{T^4AC^2AT}T^3AC^5T^2AC^6TA.$$

Repeating the operations 16, 3, 4 until the context exists leads to the following DNA strand.

$$A^4TG^3\boldsymbol{T^4AC^2}A^3TG^2\boldsymbol{T^3AC^5}A^2TG^4\boldsymbol{T^2AC^6}ATTA.$$

The modified DNA strand after removing the extra symbols is given below. By operations 1 and 2, we obtain

$$A^4TG^3C^2A^3TG^2C^5A^2TG^4C^6AT.$$

After adding and deleting the extra symbols, the above DNA strand gets modified as below.

$$A^4T\boldsymbol{G^3C^2G^5G^3C^2}A^3T\boldsymbol{G^2C^5G^7G^2C^5}A^2T\boldsymbol{G^4C^6G}^{10}\boldsymbol{G^4C^6}AT,$$

by operation 17. Deleting these extra symbols yields the following strand, by operations 5 and 6

$$A^4T\boldsymbol{G}^5A^3T\boldsymbol{G^7}A^2T\boldsymbol{G}^{10}AT.$$

In an octal number, a digit can take values ranging from 0 to 7. Hence, when the value of the result exceeds 7, a carry is said to occur. This carry has to be propagated/summed up with the value in the next higher position. By operations 18, 8 and 9, we obtain

$$\boldsymbol{G^1A^4TG^0}A^3TG^0\boldsymbol{A^2TG^2}AT.$$

When carry propagation occurs in the most significant digit of a number, then the resulting DNA strand may not begin with the base A. Hence, it is added using the operations 10, 11 and 15.

$$\boldsymbol{A^5TG^1}A^4TG^0A^3TG^0A^2TG^2AT.$$

The above process is repeated until all the octal numbers are added.

5.2 Octal Subtraction

Insertion and deletion operations necessary for performing octal subtraction are listed below.

1.$(AT,TA,B)_D$;
2.$(A^iTG^\star,T^iA,C^\star A)_D$;
3.$(A^iTG^\star,A^iTG^\star T^jAC^k,T^jAC^kA^iTG^\star)_D$;
4.$(T^jAC^kA^iTG^\star,A^iTG^\star T^jAC^k,T^{i-1}AC^\star)_D$;
5.$(A^kT,G^iC^j,G^{i-j})_D$;
6.$(G^{i-j},G^iC^j,A^kT)_D$;
7.$(A^kT,G^iC^j,C^{i-j})_D$;
8.$(C^{i-j},G^iC^j,A^kT)_D$;
9.$(A^kT,G^iA^+TC^k,G^{i-1}A^+TG^{8-k})_D$;
10.$(G^{i-1}A^+TG^{8-k},G^iA^+TC^k,A^kT)_D$;
11.$(B,A^{i+1}T,A^iT)_D$;
12.$(G,AT\lambda,AT\text{ Second DNA Strand}\lambda)_D$;
13.$(AT\text{ Second DNA Strand}\lambda,AT\lambda,\lambda)_D$;
14.$(AT\lambda,AT\text{ Second DNA Strand}\lambda)_I$;
15.$(A^iTG^\star T^jAC^k,T^jAC^kA^iTG^\star)_I$; if $i > j$
16.$(G^iC^j,G^{i-j})_I$; if $i \geq j$
17.$(G^iC^j,C^{j-i})_I$; if $i < j$
18.$(G^iA^+TC^k,G^{i-1}A^+TG^{8-k})_I$;

The detailed explanation of the above algorithm is given by the example in the following section.

5.3 Case Study

Let 173 be subtracted from 456. Representation of these two numbers using DNA bases are shown below.

$$456: A^4TG^4A^3TG^5A^2TG^6AT$$

$$173: T^4AC^1T^3AC^7T^2AC^3TA.$$

First the strand

$$A^4TG^4A^3TG^5A^2TG^6AT$$

is inserted into the test tube. Followed by it, the DNA strands of the remaining numbers are inserted into the test tube. By operations 12, 13, and 14, we obtain
$$A^4TG^4A^3TG^5A^2TG^6AT\boldsymbol{T^4AC^1T^3AC^7T^2AC^3TA}.$$
Embedding the second DNA strand after the first DNA strand is achieved by the above operations. After embedding, the digits present in the second DNA strand are percolated towards the digits of the first DNA strand. This is performed by the operations 15, 3 and 4. Repeating these operations yields the following DNA strand.

$$A^4TG^4\boldsymbol{T^4AC^1}A^3TG^5\boldsymbol{T^3AC^7}A^2TG^6\boldsymbol{T^2AC^3}ATTA.$$

Now, the extra symbols between the two operands are deleted using operations 1 and 2.

$$A^4TG^4C^1A^3TG^5C^2A^2TG^2C^3AT$$

Subtracting and deleting the extra symbols using the operations 5, 6, 7, 8, 16 and 17 results in the following DNA strand.

$$A^4T\boldsymbol{G^3}A^3T\boldsymbol{C^2}A^2T\boldsymbol{G}^3AT.$$

From the above DNA strand, it is observed that the digit in ten's place is negative and borrow from the next higher value must be done. A borrow in an octal number results in the addition of the value 8 to the digit that is borrowing. Thus, the borrow operation in octal number(base 8) is performed by subtracting the corresponding digit from 8. Thus, subtracting 2 from 8 yields 6. By operations 18, 9 and 10, we obtain

$$A^4T\boldsymbol{G^3}A^3T\boldsymbol{G^6}A^2T\boldsymbol{G^3}AT.$$

The above process is repeated until all the octal numbers in the input are subtracted.

6 Hexadecimal Arithmetic Operations

In this section, the procedure for performing hexadecimal arithmetic operations such as addition, subtraction, multiplication and division are discussed. From the procedure proposed for decimal and octal arithmetic following points are inferred. For either carry propagation or borrow operation, the value of the digit that is carried forward or borrowed is the base value of the number system. Thus the algorithm for performing arithmetic operations on hexadecimal numbers can be obtained by replacing the base value 10 or 8 by the base value 16.

Thus, the proposed algorithm can be used for performing arithmetic operations on any number systems.

Theorem 1.*The proposed algorithms for addition operation works well for all cases.*

*Proof.*The theorem can be proved by applying mathematical induction on the length of the numbers to be added. Let d and s represent a digit and a string of digits.
Basis:

Let $|s|$=1. i.e. we take two digits that need to be added. Let $d1$ and $d2$ be the two digits to be added. Then, the DNA representation for the above two digits are $A^2TG^{d1}AT$ and $T^2AC^{d2}AT$. The addition of the digits G^{d1} and C^{d2} is achieved by applying the operations numbered from 1 to 18 once.
Induction:

Let s' and s be a string of length n and $n+1$ respectively and $s = s'd$.

Assume that the theorem is true for a string of length n. i.e. the theorem is true for the string s'. Then we have to prove that it is true for a string of length $n+1$. i.e. $s'd$. Let $s1'$ and $s2'$ be the strings of length n and $d1'$ and $d2'$ be the digits that need to be concatenated with $s1'$ and $s2'$ respectively.

Let $s1' = d'_{11}d'_{12}\dots d'_{1n}$ and $s2' = d'_{21}d'_{22}\dots d'_{2n}$. The DNA representation for the strings $s1'$ and $s2'$ are given below

$$A^kTd'_{11}A^{k-1}Td'_{12}\dots A^2Td'_{1n}TA$$

and

$$T^kAd'_{21}T^{k-1}Ad'_{22}\dots T^2Ad'_{2n}TA$$

From our assumption, the theorem is true for the above two strings $s1'$ and $s2'$. Now we prove that it is true for strings of length $n+1$ i.e. for $s1$ and $s2$. We know that $s1$ and $s2$ are made by concatenating $s1'$ and $s2'$ with $d1$ and $d2$ respectively. The DNA representations for $s1$ and $s2$ are given below,

$$A^kTd'_{11}A^{k-1}Td'_{12}\dots A^2Td_1AT$$

and

$$T^kAd'_{21}T^{k-1}Ad'_{22}\dots T^2Ad_2TA$$

The above-said concatenation can be done either to the digits of s' or to the result obtained by summing $s1'$ with $s2'$. Here, the concatenation is done to the digits of s'. The sum of the above numbers can be achieved by the operations numbered from 1 to 17. The process that exists while addition can be taken care by the operation 18. Thus, the above theorem is proved.

7 Perspective

In this paper, we proposed two novel algorithms for performing arithmetic operations like addition, subtraction. These two algorithms can be used to perform the multiplication and the division operations. Also, we explained a new representation for decimal numbers in DNA computing. In Future work, we will explain the usage of these algorithms to perform a more complex mathematical operation in DNA computing.

Acknowledgement

The authors are grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References

[1] Y.Duan, G. Fu, N. Zhou, X. Sun, N. Narendra, B. Hu, Everything as a Service (XaaS) on the Cloud: Origins, Current and Future Trends, 2015 IEEE 8th International Conference on Cloud Computing. IEEE, ISBN 978-1-4673-7287-9, pp. 621–628, 2015 doi:10.1109/CLOUD.2015.88.

[2] G. von Laszewski, J. Diaz, F. Wang and G. C. Fox, "Comparison of Multiple Cloud Frameworks," 2012 IEEE Fifth International Conference on Cloud Computing, Honolulu, HI, pp. 734-741, 2012, doi: 10.1109/CLOUD.2012.104

[3] S.He, L. Guo, Y. Guo, M. Ghanem, Improving Resource Utilisation in the Cloud Environment Using Multivariate Probabilistic Models, 2012 2012 IEEE 5th International Conference on Cloud Computing (CLOUD). pp. 574–581, 2012

[4] M. Mao, M. Humphrey, A Performance Study on the VM Startup Time in the Cloud, Proceedings of 2012 IEEE 5th International Conference on Cloud Computing (Cloud2012), pp. 423, ISBN 978-1-4673-2892-0, 2012, doi:10.1109/CLOUD.2012.103.

[5] M. Abdel-Aty, Quantum information entropy and multi-qubit entanglement, Progress in Quantum Electronics, 31(1), pp. 1-49, 2007

[6] M. Abdel-Aty, An investigation of entanglement and quasiprobability distribution in a generalized Jaynes-Cummings model, Journal of Mathematical Physics 44(4), pp. 1457-1471, 2003

[7] N. Metwally, M. Abdelaty, A.-S.F Obada, Entangled states and information induced by the atom-field interaction, Optics Communications 250(1-3), pp. 148-156, 2005

[8] M. Zidan, A.-H. Abdel-Aty, M. El-shafei, M. Feraig, Y. El-Abou, H. Eleuch, M. Abdel-Aty, Quantum Classification Algorithm Based on Competitive Learning Neural Network and Entanglement Measure, Appl. Sci., 9, 1277, 2019.

[9] M. Ogihara and A. Ray, Simulating Boolean circuits on a DNA computer, Algorithmica 25:239–250, 1999.

[10] Y. Benenson, B. Gil, U. Ben-Dor, R. Adar, E. Shapiro, An autonomous molecular computer for logical control of gene expression, Nature. 429 (6990): 423–429, 2004.

[11] L. Adleman, Molecular computation of solutions to combinatorial problems, Science, 266, pp. 1021-1024, 1994.
[12] D. Ashlock and J.B. Golden, Iterated Function Systems Fractals for the Detection and Display of DNA Reading Frame, Proceedings of the 2000 Congress on Evolutionary Computation, pp. 1160–1167, 2000.
[13] Y. Brown, Arithmetic Computation in Tile Assembly Model, Theoretical Computer Science, pp. 17–31, 2007.
[14] R. Barua and J. Misra, Binary Arithmetic for DNA Computers, DNA Computing Lecture Notes in Computer Science, 2568, pp. 124–132, 2003.
[15] C.W. Dieffenbach and G.S. Dveksler, A Laboratory Manual, Cold Spring Harbor Laboratory Press, 2568, pp. 581–621, 1995.
[16] M. Daley, L. Kari, G. Gloor and R. Siromoney, Circular contextual insertions/deletions with applications to biomolecular computation, SPIRE, 2568, pp. 47–54, 1999.
[17] S. Eilenberg, Automata, Languages and Machines, Academic Press, New York, A, pp. 47–54, 1974.
[18] D. Freifelder, Molecular Biology, Narosa publishing House, New Delhi, New York, 2nd Edition, 2000.
[19] V. Gupta, S. Parthasarathy and M. J. Zaka, Arithmetic and Logic operation with DNA, In 3rd DIMACS Workshop on DNA Based Computers, New Delhi, New York, 2nd Edition, pp. 212–222, 1997.
[20] D. Giammarresi and A. Restivo, Recognizable picture languages, International Journal Pattern Recognition and Artificial Intelligence, Special Issue on Parallel Image Processing.M. Nivat, A. Saoudi and P.S.P. Wangs (Eds), 6(2-3), 241–256, 1992.
[21] D. Giammarresi and A. Restivo, Two-dimensional finite state recognizability, Fundamenta Informaticae, Special Issue : Formal Language Theory, 25(3-4), 399–422, 1996.
[22] D. Hearn and M.P. Baker, Computer Graphics, Fundamenta Informaticae, PHI, New Delhi, 2nd Edition, 2002.
[23] J.E. Hopcroft and J.D Ullman, Introduction to Automata theory Languages and computation, Narosa Publishing House, New Delhi, 2nd Edition, 1999.
[24] L. Kari, DNA computing: Arrival of Biological Mathematics, The Mathematical Intelligencer, 19, 9–22, 1997
[25] L. Kari, From Micro-soft to Bio-soft : Computing with DNA, World Scientific, Proceedings of Bio-Computing and Emergent Comp., 146–165, 1997
[26] L. Kari and M. Daley and G. Gloor and R. Siromoney, Circular Contextual insertions/deletions with applications to biomolecular computation, Proceedings of SPIRE'99,6th International symposium on string processing and information retrieval, IEEE Computing Society Press, 47–54, 1999
[27] L. Kari and G. Paun and G. Thierrin and S. Yu, At the cross roads of linguistics, DNA computing and formal language theory : characterizing RE using insertion-deletion system, DIMACS Ser. Discrete Math. Theoret. Comp. Sci., IEEE Computing Society Press, 48, 329–346, 1998
[28] K. Krithivasan and M. Sakthi Balan and R. Rama, Array Contextual Grammars, In : C.Martin-Vide and G.H Paun Eds, Recent topics in Mathamatical and Computational Linguistics, 48, 154–168, 2000
[29] L. Kari and G. Thierrin, Contextual insertions/deletions and computability, Information and computation, Recent topics in Mathamatical and Computational Linguistics, 131, 47–61, 1996
[30] S.A. Krawetz and D.D. Womble, Introduction to Bioinformatics, Humana Press, New Jersey, 2003
[31] B. Lewin, Genes VII, Oxford University Press, New York, 2000
[32] L. Kari, On insertions and deletions in formal languages, University of Turku, Finland, 1991
[33] Lila Kari and Petr Sosik, On the weight of universal insertion grammars, Theoretical Computer Science, 264–270, 2008
[34] G.S. Miglani, Basic Genetics, Narosa publishing House, New Delhi, 2000
[35] A. Murugan and K.S. Easwarakumar, DNA Algorithm for Fractal Image Generation, Tech. Report. Department of Computer Science & Engg., Anna University, Chennai, India
[36] A. Murugan and K.S. Easwarakumar, Transposition based contextual insertion on linear DNA strands with deletion precedence, International Journal of Computer Mathematics, 81(6), 647–660, 2004
[37] MIN Yong, JIN Xiao-gnag, SU Xian-Chuang and PENG Bo, Minimizing of the only-insertion insdel systems, Journal of Zhejiang University SCIENCE, 1021–1025, 2005
[38] G. Paun and G. Rozenberg and A. Salomaa, DNA Computing: New Computing Paradigms, Springer-Verlag, Berlin, 1998
[39] G.Rozenberg and H.Spaink, DNA computing by blocking, Theoretical Computer Science, 292, pp. 653–665, 2003
[40] Z. F. Qiu and M. Liu, Arithmetic and Logic Operations for DNA computers, In proceeding of the second IASTED International conference on Parallel and Distributed Computing and Networks, pp. 481-486, 1998
[41] G. Rozenberg and A. Salomaa, Lindenmayer Systems, In proceeding of the second IASTED International conference on Parallel and Distributed Computing and Networks, USA, 1992
[42] A. Salomaa, Formal Languages, Academic Press, New York, 1973
[43] R. Siromoney and K. Krithivasan, Parallel context-free languages, Inform. and Control, 24, pp. 155-162, 1974
[44] G. Siromoney, R. Siromoney and K. Krithivasan, Picture Languages with Array rewriting Rules, Inform. and Control, 22(5), pp. 447–470, 1973
[45] G. Siromoney, R. Siromoney and K. Krithivasan, Abstract Families of Matrices and Picture Languages, Computer Graphics and Image Processing, 1(3), 1972
[46] R. Siromoney, K. Subramanian and K. Rangarajan, Parallel/Sequential rectangular arrays with tables, Intern. Journ. Computer Math., 6, pp. 143–158, 1977
[47] J. Wang and Y. Huang, Decimal Integer Multiplication based on Molecular Beacons, Journal of Networks, 8(12), pp. 2762–2768, 2013
[48] T. Zingel, Formal models of DNA computing : A Survey, Proc. Estonian Acad. Sci. Phys. Maths., 49(2), pp. 90–99, 2000
[49] X. Zhang et.al, Arithmetic computation using self-assembly of DNA tiles: subtraction and division, Science Direct., pp. 377–388, 2009

V. Sudha received her BE degree in Computer Science and Engineering from VMKV Engineering College, Salem and ME degree in Computer Science and Engineering from CEG Campus, Anna University. She is currently pursuing her PhD in Anna University in the area of Theoretical Computer Science.

K. S. Easwarakumar is a Professor at the Department of Computer Science and Engineering at Anna University, Chennai. He received his M. Tech in Computer and Information Sciences from Cochin University of Science and Technology, Cochin and PhD in Computer Science and Engineering from Indian Institute of Technology, Madras. He was the Head of the Department of Computer Science, in Anna University, Chennai for three years. For the past 25 years, he has dedicated himself to academia in Computer Science. His research interests include parallel and distributed computing, data structures, graph algorithms, parallel algorithms, computational geometry, theoretical computer science, molecular computing, and pattern recognition.