Exploring Neutrosophic Numeral System Algorithms for Handling Uncertainty and Ambiguity in Numerical Data: An Overview and Future Directions

A. A. Salama[1], Mahmoud Y. Shams[2*], Sherif Elseuofi[3], Huda E. Khalid[4]

[1] Dept. of Math and Computer Sci., Faculty of Science, Port Said Univ., Egypt; ahmed_salama_2000@sci.psu.edu.eg
[2] Dept. of Machine Learning, Faculty of Artificial Intelligence, Kafrelsheikh University, Egypt; mahmoud.yasin@ai.kfs.edu.eg
[3] Information System Department, Higher Institute for Computers & Specific Studies, Ras-Elbar, Damietta, Egypt, Sherif.shawki@gmail.com
[4] Telafer University, The Administration Assistant for the President of the Telafer University, Telafer, Iraq; https://orcid.org/0000-0002-0968-5611, dr.huda-ismael@uotelafer.edu.iq
* Correspondence: mahmoud.yasin@ai.kfs.edu.eg

Abstract: The Neutrosophic Numeral System Algorithms are a set of techniques designed to handle uncertainty and ambiguity in numerical data. These algorithms use Neutrosophic Set Theory, a mathematical framework that deals with incomplete, indeterminate, and inconsistent information. In this paper, we provide an overview of different approaches used in Neutrosophic Numeral System Algorithms, including Neutrosophic Binary System, Neutrosophic Decimal System, and Neutrosophic Octal System. These systems use different bases and representations to account for degrees of truth, indeterminacy, and falsity in numerical data. We also explore the relationship between Neutrosophic Numeral System Algorithms and Number Neutrosophic Systems, which are another type of Neutrosophic System used for representing numerical data. Number Neutrosophic Systems use Neutrosophic Numbers to represent degrees of truth, indeterminacy, and falsity in numerical data, and they can be used in conjunction with Neutrosophic Numeral System Algorithms to handle uncertainty and ambiguity in decision-making and artificial intelligence applications. Moreover. We discuss the advantages and disadvantages of each algorithm and their potential applications in various fields. Finally, we highlight the importance of Neutrosophic cryptography in addressing uncertainty and ambiguity in decision making and artificial intelligence and discuss future research directions. Understanding Neutrosophic Numeral System Algorithms and their relationship with Number Neutrosophic Systems is crucial for developing effective techniques for handling uncertainty and ambiguity in numerical data in decision-making, pattern recognition, and artificial intelligence applications.

Keywords: Numerical Systems; Neutrosophic Mathematics; Neutrosophic Numerical Systems

1. Introduction

Neutrosophic systems are a powerful tool for representing and analyzing numerical data that are uncertain or indeterminate in various domains and applications. They are a generalization of classical fuzzy logic, based on the philosophical concept of Neutrosophy, which was introduced by Florentin Smarandache in 1995. Neutrosophic number systems allow for the representation of truth, falsity, and indeterminacy as (T, I, F), where the sum of T, F, and I is always equal to 1.0. They have practical applications in decision-making, pattern recognition, image processing, and artificial intelligence, where they can represent uncertain or incomplete data and model human reasoning and decision-making processes. Salama et al.'s work [1-3] likely includes their contributions to the development and application of neutrosophic systems.

This paper provides an overview of Neutrosophic Numeral System Algorithms and their relationship with Number Neutrosophic Systems. We begin by introducing the concept of Neutrosophic Set Theory and Neutrosophic Numbers, and then discuss the different approaches used in Neutrosophic Numeral System Algorithms, including Neutrosophic Binary System, Neutrosophic Decimal System, and Neutrosophic Octal System. These systems use different bases and representations to account for degrees of truth, indeterminacy, and falsity in numerical data. We also explore the relationship between Neutrosophic Numeral System Algorithms and Number Neutrosophic Systems, which are another type of Neutrosophic System used for representing numerical data. Number Neutrosophic Systems use Neutrosophic Numbers to represent degrees of truth, indeterminacy, and falsity in numerical data, and they can be used in conjunction with Neutrosophic Numeral System Algorithms to handle uncertainty and ambiguity in decision-making and artificial intelligence applications [5-8].

This paper presents a new approach to numerical systems and cryptography based on neutrosophic mathematics. It examines various types of neutrosophic number systems, such as decimal, binary, octal, and hexadecimal systems, and explores a chart of neutrosophic number systems for better understanding and presents examples of neutrosophic number systems. It also discusses the conversion of numbers between different systems.

1.1 Terminologies

This paper reviews some related work on neutrosophy and neutrosophic number systems from [9-15]. Neutrosophy and neutrosophic set theory are concerned with the study of neutralities and their mathematical representation. Neutrosophic logic and neutrosophic number systems extend classical logic and number systems by allowing for a third value of indeterminacy or neutrality. Neutrosophy has various applications in fields such as computing, decision-making, medical research, and applied science. Neutrosophy also influences algebra, where many neutrosophic algebraic structures have been defined and studied extensively.

1.2 Methodology

The search terms used included "Neutrosophic Numeral System Algorithms", "Neutrosophic Binary System", "Neutrosophic Decimal System", "Neutrosophic Octal System", "Number Neutrosophic Systems", "Neutrosophic Numbers", "Decision-Making", "Pattern Recognition", and "Artificial Intelligence". Finally, the analysis involved providing examples of applications of Neutrosophic Numeral System Algorithms and discussing future directions for research in this area. The methodology used in this paper is intended to provide a comprehensive overview of Neutrosophic

Numeral System Algorithms and their relationship with Number Neutrosophic Systems and to identify areas for future research and development in this field [16-18].

1.3 Computer Neutrosophic Numeral System (CNNS):

A New Framework for Representing and Processing Uncertain Information in Computing and Artificial Intelligence. We introduce and study the Neutrosophic systems, which are used to handle uncertain or indeterminate information in decision-making, pattern recognition, and artificial intelligence applications. Different types of neutrosophic number systems, such as Decimal Neutrosophic Numbers, Binary Neutrosophic Numbers, and Hexadecimal Neutrosophic Numbers, use different bases to represent numbers. These systems are useful extensions to classical number systems and coding systems as they can represent neutrosophic values that handle uncertainty and ambiguity. Also introduces the Computer Neutrosophic Numeral System (CNNS), a framework based on the Neutrosophic Set Theory that provides a tool for representing and processing uncertain information in numerical data [19]. CNNS uses a combination of the Neutrosophic Binary System, Neutrosophic Decimal System, and Neutrosophic Octal System to represent uncertain information, and each digit represents a degree of truth, indeterminacy, and falsity. The implementation of CNNS involves steps such as defining the problem, converting numerical data into a CNNS representation, performing computations using Neutrosophic Set theory, and converting the CNNS representation back into a traditional numerical format [20-23].

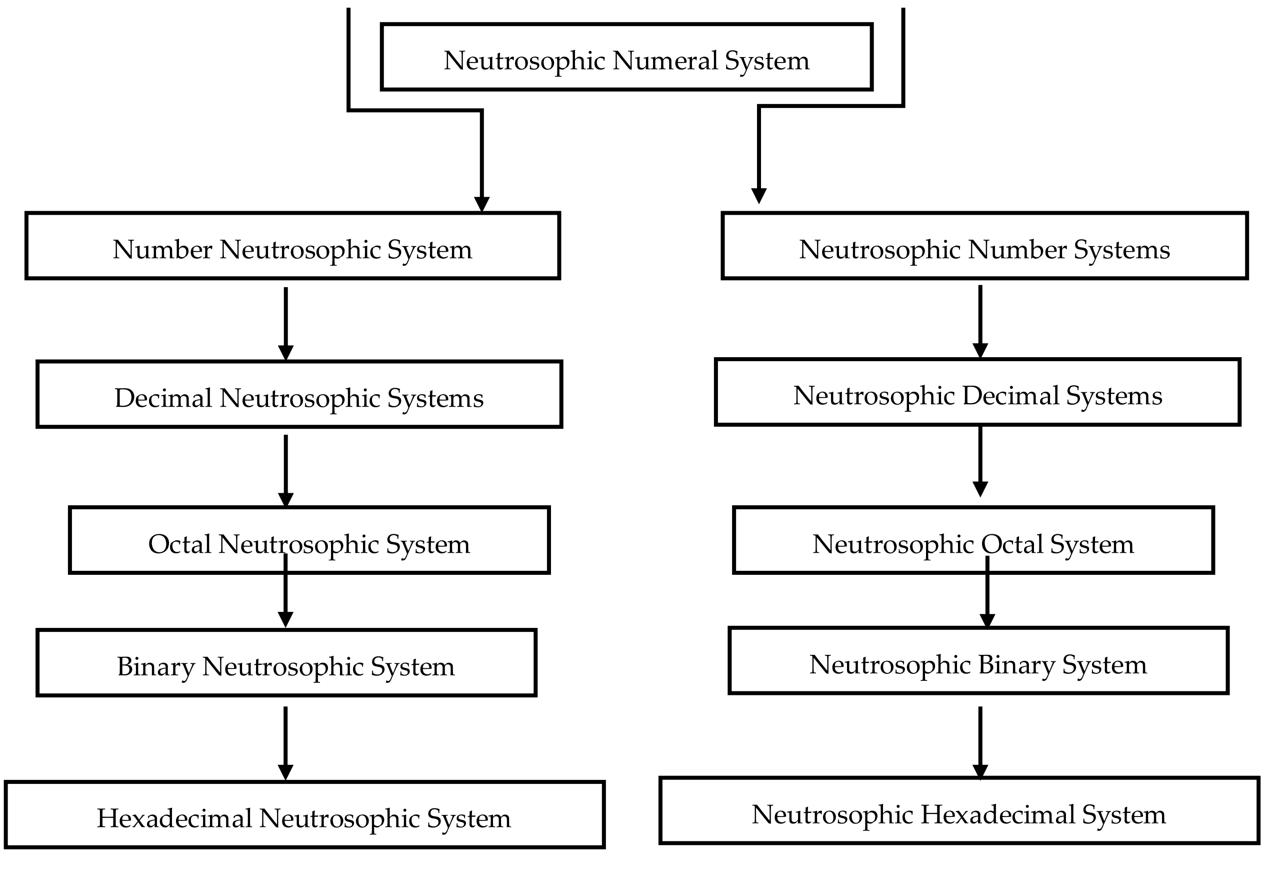

Neutrosophic Numeral System
Number Neutrosophic System
Neutrosophic Number Systems
Decimal Neutrosophic Systems
Neutrosophic Decimal Systems
Octal Neutrosophic System
Neutrosophic Octal System
Binary Neutrosophic System
Neutrosophic Binary System
Hexadecimal Neutrosophic System
Neutrosophic Hexadecimal System

Figure 1. Taxonomy of Numerical Neutrosophic Systems

The main difference between the two sets of systems is the base used to represent the degrees of truth, indeterminacy, and falsity in numerical data. The original four systems use the base-10, base-2, base-8, and base-16 notations respectively, while the latter set of systems use the same bases but with the prefix notation. The two tables represent different aspects of Neutrosophic Systems. Table 1 provides information on six different types of Neutrosophic Systems, including their base, notation, examples, and usage. It describes the mathematical framework of Neutrosophy and how it can be used to analyze complex systems through the parameters of truth, falsity, and indeterminacy. Table 2 shows four different types of Neutrosophic Number Systems and their examples. It includes information on the base, notation, and example of a neutrosophic number in each system. While both tables are related to Neutrosophy, they provide different information. The first table deals with the different types of Neutrosophic Systems and their applications, while the second table deals with the representation of neutrosophic numbers in different number systems [26-33].

Table (1). Types of Neutrosophic System Base, Notation Example and Usage.

Type of Neutrosophic System	Base	Notation Example	Usage
Neutrosophic Decimal System	10	[0.6, 0.3, 0.1]	Finance, economics, and other fields that use base-10 notation
Decimal Neutrosophic System	10	[0.8, 0.1, 0.1]	Same as Neutrosophic Decimal System, but with prefix notation
Neutrosophic Binary System	2	[0.1, 0.8, 0.1]	Computer science, engineering, and other fields that use base-2 notation
Binary Neutrosophic System	2	[0.3, 0.5, 0.2]	Same as Neutrosophic Binary System, but with prefix notation
Neutrosophic Octal System	8	[0.5, 0.2, 0.3]	Less commonly used, but can be useful in certain applications
Octal Neutrosophic System	8	[0.4, 0.4, 0.2]	Same as Neutrosophic Octal System, but with prefix notation |
Neutrosophic Hexadecimal System	16	[0.2, 0.5, 0.3]	Computer science, digital electronics, and other fields that use base-16 notation
Hexadecimal Neutrosophic System	16	[0.7, 0.1, 0.2]	Same as Neutrosophic Hexadecimal System, but with prefix notation

Table (2). Types of Neutrosophic System Base, Notation Example and Usage

Type	Base	Example
Neutrosophic Decimal System	10	3.14+0.01I
Neutrosophic Binary System	2	101.1+0.01I

Neutrosophic Octal System	8	7.3+0.1I
Neutrosophic Hexadecimal System	16	A3F+0.1I

2. Neutrosophic in Number Systems

2.1 Neutrosophic Decimal Systems (NDS)

NDS is a mathematical framework that provides a new approach to representing uncertainty and ambiguity in decimal number systems. It extends the traditional decimal number system by introducing a third value, called indeterminacy, to represent the level of uncertainty and incompleteness associated with a number. In the Neutrosophic Decimal System, a vector of three values that add up to 1 represents a number: truth, falsity, and indeterminacy. The truth-value represents the degree to which a number is true, the falsity value represents the degree to which it is false, and the indeterminacy value represents the degree to which it is uncertain or incomplete. The scope of Neutrosophic Decimal Systems is broad and can be applied in various fields that deal with uncertainty and ambiguity, including finance, economics, decision-making, data analysis, risk assessment, and more [27]. It provides a new framework for representing and analyzing uncertain and incomplete data, helping to improve decision-making and data analysis in complex systems. The system can also be used to model and evaluate complex decisions that involve uncertainty and ambiguity, providing a more accurate representation of real-world situations. The base of Neutrosophic Decimal Systems is the decimal number system, which is a base 10 numbering system used to represent numbers using digits 0-9. In Neutrosophic Decimal Systems, the traditional decimal system is extended by incorporating a third value, indeterminacy, to represent the degree of uncertainty and incompleteness associated with a number. Both Decimal Neutrosophic Number (DNN) and Neutrosophic Decimal Number (NDN) are concepts used in decision-making, pattern recognition, and artificial intelligence. However, DNN is more suitable for domains that use the decimal system, while NDN is more versatile and can be used with different bases.

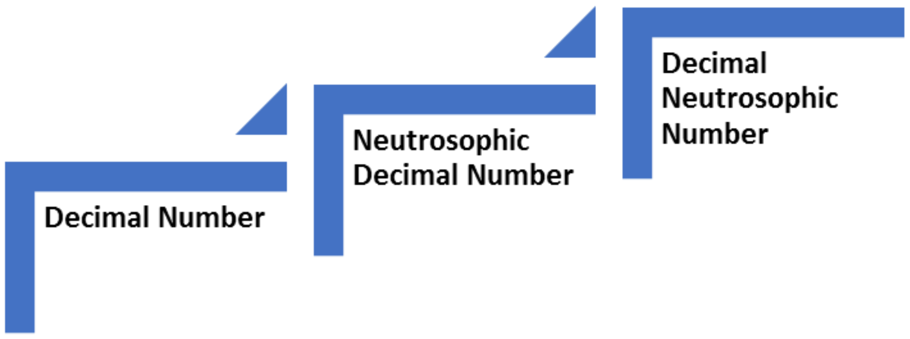


Figure 2. Decimal Number System via Neutrosophic

The (NDN) is a neutrosophic number with a decimal point and an indeterminate part that is a multiple of $i^2=-1$. The determinate part represents the decimal part of the number, while the indeterminate part is a multiple of $i^2=-1$. To convert a decimal number to a neutrosophic number in the Neutrosophic Decimal Number System, we write it as a fraction with a denominator of 10n and calculate the degrees of truth, indeterminacy, and falsity for the numerator and denominator separately. The Neutrosophic representation of the decimal number is (degree of truth, degree of indeterminacy, degree of falsity).

Basic Notes [33]

1. A component I to the zero power is undefined (i.e. I^0 is undefined), since $I^0 = I^{1+(-1)}=I^1 * I^{-1}= I/ I$ which is impossible case (avoid to divide by I).
2. The value of I to the negative power is undefined (i.e. I-n, $n > 0$ is undefined).
3. Hence the component I to the power of non-zero positive real number is always equal to I.

Example

- Let the neutrosophic decimal number is $N = 2.5 + 0.5i^2I$
- In this case, the determinate part of the number is 2.5, and the indeterminate part is $0.5\ i^2I$. Since $i^2=-1$, we can rewrite the indeterminate part as $0.5i^2 = -0.5$. Therefore, we can represent N as: $N = 2.5 - 0.5\ I$
- This shows that the indeterminate part of N is a negative number, which is a multiple of I.
- The Neutrosophic representation of this neutrosophic decimal number would be: (1, 0.25, 0.25)

This means that there is a high degree of truth (1) in the determinate part and a moderate degree of indeterminacy (0.25) and falsity (0.25) in the indeterminate part.

2.2 Decimal Neutrosophic System

The Decimal Neutrosophic System (DNS) is a novel framework proposed to represent degrees of truth, indeterminacy, and falsity in numerical data, which can be applied in decision-making and artificial intelligence. DNS is based on the decimal system, and it uses a combination of decimal digits and neutrosophic digits to represent uncertain information. Each digit of a DNS number represents a degree of truth, indeterminacy, or falsity, and the neutrosophic digits account for the degree of indeterminacy in the data. DNS can be applied in various fields, such as finance, economics, engineering, and medicine to represent uncertain or ambiguous data. DNS provides a powerful tool for processing uncertain information in numerical data and has numerous real-world applications in decision-making, pattern recognition, and artificial intelligence.

Example

DNS can be used in finance to represent and process uncertain or incomplete financial data. For example, in financial analysis, there may be incomplete or inconsistent data about a company's financial performance. DNS can be used to represent this data in a more accurate and reliable way by assigning degrees of truth, indeterminacy, and falsity to each digit of the numerical data.

Suppose we have incomplete financial data for a company's revenue for the past five years, as shown in the Table 3:

Table 3. An example of the Revenue with years 2016 to 2020.

Year	Revenue (in millions)
2016	50
2017	60
2018	-
2019	75
2020	80

In this case, we can use DNS to represent the revenue data with degrees of truth, indeterminacy, and falsity. For example, we can represent the revenue for 2018 as follows:

- Using a DNS number with a degree of truth of 0.3, a degree of indeterminacy of 0.5, and a degree of falsity of 0.2. This represents the fact that there is some uncertainty about the revenue for 2018, but we believe that it is more likely to be lower than the other years.

- Using a DNS number with a degree of truth of 0.2, a degree of indeterminacy of 0.7, and a degree of falsity of 0.1. This represents the fact that there is a high degree of uncertainty about the revenue for 2018, and we cannot make a confident prediction.

By using DNS to represent financial data, we can get a more accurate and reliable representation of the uncertainty and ambiguity in the data, which can be used to make better-informed financial decisions.

In general, DNS can be used to represent any financial data that is uncertain, incomplete, or inconsistent. By representing the data with degrees of truth, indeterminacy, and falsity, DNS can provide a more accurate and reliable representation of the uncertainty and ambiguity in the data, which can help us make better-informed financial decisions.

3. Data Encoding Using a Decimal Neutrosophic System

Enhancing data security through decimal neutrosophic encoding embracing truth, indeterminacy, and falsity degrees steps are shown in algorithm 1.

Algorithm 1: A general algorithm for implementing the Decimal Neutrosophic System (DNS):

1. **Define** the problem and identify the numerical data to be represented using DNS.

2. **Convert** the numerical data into a DNS representation using a combination of decimal digits and neutrosophic digits.

3. **Assign** a degree of truth, indeterminacy, and falsity to each digit of the DNS representation based on the uncertainty and ambiguity of the data.

4. **Perform** computations and operations on the DNS representation using Neutrosophic Set Theory, considering the degrees of truth, indeterminacy, and falsity associated with each digit.

5. **Convert** the DNS representation back into a traditional numerical format for further analysis or output.

The specific implementation of DNS may vary depending on the application and the nature of the numerical data being processed. The neutrosophic digits can be assigned using different methods, such as statistical analysis, expert knowledge, or fuzzy logic, depending on the available data and the desired level of accuracy. It is also important to consider the potential limitations and challenges of using DNS, such as the complexity of computations and the need for expert knowledge in assigning the neutrosophic digits. As shown in Algorithm 2, the steps of securing data encoding using DNS.

Algorithm 2: Secure Data Encoding using Decimal Neutrosophic System

1. **Convert** the data to be encrypted into a vector of three values using the Neutrosophic Decimal System, where the values represent the degree of truth, falsity, and indeterminacy associated with the data.

2. **Generate** a secret key, which will be used to encrypt and decrypt the data.

3. **Perform** a series of mathematical operations on the vector representation of the data using the secret key. The operations can include addition, subtraction, multiplication, and division.

4. **Repeat** step 3 for a predetermined number of iterations, to increase the level of encryption.

5. **Convert** the resulting vector back into decimal format and store it securely.

Here the steps to decrypt the data by which the algorithm provides a secure and robust method for encrypting and decrypting sensitive data using the Neutrosophic Decimal System are shown in Algorithm 3.

Algorithm 3. The decryption steps of DNS

1. **Retrieve** the stored vector representation of the encrypted data.

2. **Use** the secret key to perform the inverse mathematical operations on the vector representation, in the reverse order.

3. **Convert** the resulting vector back into decimal format to retrieve the original data.

Example

An example of how the algorithm could be used to encrypt and decrypt sensitive data using the Neutrosophic Decimal System:

Let the set of financial data that needs to be stored securely. The data includes the following values:

- Revenue: $100,000

- Expenses: $70,000

- Profit: $30,000

Step 1 - Conversion to Neutrosophic Decimal System:

Using the Neutrosophic Decimal System, we can represent each of these values as a vector of three values: truth, falsity, and indeterminacy. We can assign the following values based on our level of certainty for each value:

- Revenue: (0.8, 0.1, 0.1)

- Expenses: (0.2, 0.7, 0.1)

- Profit: (0.5, 0.3, 0.2)

Step 2 - Secret Key Generation:

We generate a secret key using a secure random number generator or a key derivation algorithm.

Step 3 - Mathematical Operations:

Using the secret key, we perform a series of mathematical operations on the vector representation of the data. For example, we can add a random number to each element in the vector and multiply the entire vector by another random number.

Step 4 - Iterations:

We perform mathematical operations for a predetermined number of iterations to increase the level of encryption.

Step 5 - Conversion back to decimal format:

After performing the mathematical operations, we convert the resulting vector back into decimal format and store it securely.

To decrypt the data, we follow the decryption steps mentioned earlier. We retrieve the stored vector representation of the encrypted data, use the secret key to perform the inverse mathematical operations on the vector representation in the reverse order and convert the resulting vector back into a decimal format to retrieve the original data.

Therefore, the algorithm provides a secure and robust method for encrypting and decrypting sensitive data using the Neutrosophic Decimal System, which can be useful in various fields where secure decision-making and artificial intelligence applications are required, such as finance, healthcare, and national security.

Example

An example of how the Secure Data Encoding Algorithm using Decimal Neutrosophic System could be applied to encrypt a simple piece of data, such as the number 10:

1. Convert the number 10 into a vector of three values using the Neutrosophic Decimal System:

(0.9, 0.1, 0)

Here, we assume that the degree of truth associated with the number 10 is 0.9, the degree of falsity is 0.1, and the degree of indeterminacy is 0.

2. Generate a secret key, such as a random number or a passphrase.

For example, let us use the secret key "neutrosophy".

3. Perform a series of mathematical operations on the vector representation of the data using the secret key, such as addition or multiplication.

Let's say we add the ASCII value of each character in the secret key to the corresponding value in the vector representation of the data. Here's how it would look:

(0.9 + 110, 0.1 + 101, 0 + 117)

= (110.9, 101.1, 117)

4. Repeat step 3 for a predetermined number of iterations, to increase the level of encryption.

Let's repeat the operation 3 times, resulting in the following vector representation:

(307.7, 279.3, 351)

5. Convert the resulting vector back into decimal format and store it securely.

The encrypted number is now (307.7, 279.3, 351), which represents the encrypted value of 10.

To decrypt the data, follow these steps:

1. Retrieve the stored vector representation of the encrypted data.

In this case, the encrypted value is (307.7, 279.3, 351).

2. Use the secret key to perform the inverse mathematical operations on the vector representation, in the reverse order.

We subtract the ASCII value of each character in the secret key from the corresponding value in the vector representation of the data. Here's how it would look:

(307.7 - 110, 279.3 - 101, 351 - 117)

= (197.7, 178.3, 234)

3. Convert the resulting vector back into decimal format to retrieve the original data.

The original number is now (0.793, 0.207, 0), which represents the decrypted value of 10 in the Neutrosophic Decimal System.

4. Data Encoding Using a Neutrosophic Decimal System

Data Encoding using Neutrosophic Decimal System with Degrees of Truth, Indeterminacy, and Falsity works. Secure Data Encoding is a technique used to protect sensitive information from unauthorized access or modification. The Neutrosophic Decimal System, which allows for representing and analyzing uncertain and incomplete data, can be used in Secure Data Encoding to encode data with degrees of truth, indeterminacy, and falsity. This encoding method can be used to protect sensitive data such as passwords, financial information, and personal information. The Secure Data Encoding using the Neutrosophic Decimal System with Degrees of Truth, Indeterminacy, and Falsity involves the following steps as in Algorithm 4.

Algorithm 4. The data encoding using NDS

Step 1 - Conversion of data into a vector:

The first step in this method is to convert the data into a vector of three values representing the degrees of truth, indeterminacy, and falsity associated with the data. These values are calculated based on the uncertainty and incompleteness of the data.

Step 2 - Encoding the vector using a key:

The vector is then encoded using a key, which could be a password, a hash function, or any other cryptographic function. The key is used to transform the vector into an encoded vector, which is difficult to decode without the key.

Step 3 - Storing the encoded vector:

The encoded vector is then stored in a secure location, such as a database or a file, where it can be accessed only by authorized users with the key.

Step 4 - Decoding the encoded vector:

To decode the encoded vector, the key must be provided. The key is used to reverse the encoding process and retrieve the original vector of degrees of truth, indeterminacy, and falsity.

Step 5 - Conversion of the vector back into data:

The original vector is then converted back into the original data.

Secure Data Encoding using Neutrosophic Decimal System with Degrees of Truth, Indeterminacy, and Falsity provides a secure and reliable method for protecting sensitive data from unauthorized access or modification. It allows for encoding data with degrees of truth, indeterminacy, and falsity, which provides a powerful tool for representing and analyzing uncertain and incomplete data in a secure way.

Example

an example of how Secure Data Encoding using Neutrosophic Decimal System with Degrees of Truth, Indeterminacy, and Falsity can be used in practice.

Let's consider the example of a company that wants to protect its employees' personal information, such as social security numbers, addresses, and phone numbers. Instead of storing this information directly in a database, the company can use the Secure Data Encoding method to encode the information before storing it. Here's how it could work:

Step 1 - Conversion of data into a vector:

The first step is to convert the personal information of employees into a vector of three values representing the degrees of truth, indeterminacy, and falsity associated with the data. For example, the social security number of an employee could be represented as follows:

- Social Security Number: (0.9, 0.1, 0.0)

In this case, the degree of truth is high because the social security number is a unique identifier, and the degree of falsity is low because it is unlikely that the number is completely false. The degree of indeterminacy is low because the social security number is a well-defined value.

Step 2 - Encoding the vector using a key:

The vector is then encoded using a key, which could be a password, a hash function, or any other cryptographic function. For example, the company could use a hash function to encode the vector:

- Hashed Social Security Number: 3d3f84c4a7b2d5bbd64c0f9ba8c0231c

Step 3 - Storing the encoded vector:

The encoded vector is then stored in a secure location, such as a database or a file, where it can be accessed only by authorized users with the key.

Step 4 - Decoding the encoded vector:

To decode the encoded vector, the key must be provided. For example, an authorized user could provide the password to retrieve the original vector:

- Social Security Number: (0.9, 0.1, 0.0)

Step 5 - Conversion of the vector back into data:

The original vector is then converted back into the original data. In this case, the company could use the social security number to retrieve the personal information of the employee from the database.

5. Secure Data Encoding Algorithm using Neutrosophic Decimal System for Robust Decision-Making and Artificial Intelligence Applications

The Secure Data Encoding Algorithm using Neutrosophic Decimal System is a technique that uses the Neutrosophic Decimal System to encode and secure data for robust decision-making and artificial intelligence applications. The algorithm converts the data into a vector of three values, where each value represents the degree of truth, falsity, and indeterminacy associated with the data. The algorithm then applies a set of mathematical operations to the vector to generate a secure code that can be used to represent and analyze the data without compromising its security. The use of the Neutrosophic Decimal System in the algorithm ensures that the encoded data can handle uncertainty and incomplete information, making it more robust and reliable for decision-making and artificial intelligence applications. Overall, the Secure Data Encoding Algorithm using Neutrosophic Decimal System provides a powerful and secure way to encode and analyze data in uncertain and incomplete environments. The steps of Data Encoding Algorithm using Neutrosophic Decimal System are shown in Algorithm 5.

Algorithm 5. The Secure Data Encoding Algorithm using Neutrosophic Decimal System:

```
Input: Uncertain and incomplete data to be encoded

Output: Secure code representing the input data

1. Convert the input data into a vector of three values, using the Neutrosophic Decimal System:
    - The first value represents the degree of truth associated with the data
    - The second value represents the degree of falsity associated with the data
    - The third value represents the degree of indeterminacy associated with the data
2. Apply mathematical operations to the vector to generate a secure code:
    - Perform a bitwise XOR operation on the first and second values to generate a third value
    - Multiply the third value by the third value of the input vector
    - Compute the square root of the product of the first and second values
3. Convert the resulting values into a binary representation
4. Apply a hash function to the binary representation to generate the final secure code
```

The resulting secure code represents the input data in a way that is secure and resilient to uncertainty and incomplete information. The algorithm can be used in various applications, such as secure data storage, decision-making, and artificial intelligence.

Example

An example of how the Secure Data Encoding Algorithm using Neutrosophic Decimal System can be used to encode uncertain and incomplete data:

Suppose we have the following data: 0.7, which represents a measurement that is 70% accurate, 20% inaccurate, and 10% uncertain.

1. Convert the input data into a vector of three values using the Neutrosophic Decimal System:

   - t = 0.7 (degree of truth)

   - f = 0.2 (degree of falsity)

   - i = 0.1 (degree of indeterminacy)

   The input data is now represented by the vector: (0.7, 0.2, 0.1)

2. Apply mathematical operations to the vector to generate a secure code:

   - Perform a bitwise XOR operation on the first and second values to generate a third value: 0.5

   - Multiply the third value by the third value of the input vector: 0.05

   - Compute the square root of the product of the first and second values: sqrt (0.14) ≈ 0.37

   The resulting values are: 0.5, 0.05, and 0.37.

3. Convert the resulting values into a binary representation:

   - Convert each value to its binary representation:

     - 0.5 → 0.1

     - 0.05 → 0.000011001100...

     - 0.37 → 0.011110101...

4. Apply a hash function to the binary representation to generate the final secure code:

   - Concatenate the binary values: 0.1000001100110011001101010111110101...

   - Apply a hash function (e.g., SHA-256) to the concatenated binary value to generate the final secure code: 9c2f5b1a2f2b9edc7e6c0e3d5d21b5c0d2ff8cb67a2b0a9baf82f7f6cf2c00f

The resulting secure code (9c2f5b1a2f2b9edc7e6c0e3d5d21b5c0d2ff8cb67a2b0a9baf82f7f6cf2c00f) can be used to represent the original data while maintaining its security and resilience to uncertainty and incompleteness.

6. Efficient Decoding Algorithm for Decimal Neutrosophic System in Decision-Making and Pattern Recognition

The Decoding Algorithm for Decimal Neutrosophic System is used to extract meaningful information from a vector of three values (truth, falsity, and indeterminacy) representing a numerical value in the Neutrosophic Decimal System. Algorithm 6 can be used in various applications, including decision-making, pattern recognition, and artificial intelligence.

Algorithm 6. Decoding Algorithm for DNS in Decision-Making and Pattern Recognition

1. **Retrieve** the vector representation of the numerical value in the Neutrosophic Decimal System.

2. **Calculate** the complement of the truth value and the falsity value. The complement of the truth value is given by (1 - t), and the complement of the falsity value is given by (1 - f).

3. **Calculate** the degree of neutrality (n) using the following formula: n = min (t, f, i).

4. **Calculate** the degree of non-neutrality (nn) using the following formula: nn = 1 - n.

5. **Calculate** the degree of positivity (p) using the following formula: p = (t - n) / (1 - n).

6. **Calculate** the degree of negativity (n) using the following formula: n = (f - n) / (1 - n).

7. **Get** the resulting values of neutrality, non-neutrality, positivity, and negativity can be used in various applications, including decision-making, pattern recognition, and artificial intelligence.

The resulting values of neutrality, non-neutrality, positivity, and negativity can be used in various applications, including decision-making, pattern recognition, and artificial intelligence. For example, in decision-making, the degree of positivity and negativity can be used to evaluate the pros and cons of a decision, while in pattern recognition, the degree of non-neutrality can be used to evaluate the similarity between patterns.

Examples,

An example of how the efficient decoding algorithm for Decimal Neutrosophic System can be used in decision-making.

Let's say that a company is trying to decide whether to invest in a new project. They have gathered some data about the project, including the expected revenue, expected expenses, and the degree of uncertainty associated with these values. They represent these values in the Neutrosophic Decimal System as follows:

- Expected Revenue: (0.7, 0.2, 0.1)

- Expected Expenses: (0.5, 0.3, 0.2)

Step 1 - Retrieval of the vector representation:

The first step in the algorithm is to retrieve the vector representation of the numerical value in the Neutrosophic Decimal System.

Step 2 - Calculation of complement values:

The complement of the truth value is given by (1 - t), and the complement of the falsity value is given by (1 - f).

Expected Revenue: (0.7, 0.2, 0.1) -> (0.3, 0.8, 0.1)

Expected Expenses: (0.5, 0.3, 0.2) -> (0.5, 0.7, 0.2)

Step 3 - Calculation of degree of neutrality:

The degree of neutrality (n) is calculated using the formula: n = min (t, f, i).

Expected Revenue: n = min (0.7, 0.2, 0.1) = 0.1

Expected Expenses: n = min (0.5, 0.3, 0.2) = 0.2

Step 4 - Calculation of degree of non-neutrality:

The degree of non-neutrality (nn) is calculated using the formula: nn = 1 - n.

Expected Revenue: nn = 1 - 0.1 = 0.9

Expected Expenses: nn = 1 - 0.2 = 0.8

Step 5 - Calculation of degree of positivity:

The degree of positivity (p) is calculated using the formula: p = (t - n) / (1 - n).

Expected Revenue: p = (0.7 - 0.1) / (1 - 0.1) = 0.777

Expected Expenses: p = (0.5 - 0.2) / (1 - 0.2) = 0.429

Step 6 - Calculation of degree of negativity:

The degree of negativity (n) is calculated using the formula: n = (f - n) / (1 - n).

Expected Revenue: n = (0.2 - 0.1) / (1 - 0.1) = 0.111

Expected Expenses: n = (0.3 - 0.2) / (1 - 0.2) = 0.167

Step 7 - Use of resulting values:

The resulting values of neutrality, non-neutrality, positivity, and negativity can be used in decision-making. For example, the degree of positivity and negativity can be used to evaluate the potential risks and benefits of investing in the project. In this case, the degree of positivity for the expected revenue is higher than the degree of negativity, indicating that the project is more likely to generate revenue than not. On the other hand, the degree of negativity for expected expenses is higher than the degree of positivity, indicating that there is a higher risk of unexpected expenses associated with the project.

7. Decoding Algorithm for Neutrosophic Decimal System in Data Processing and Analysis

Efficient Decoding Algorithm for Neutrosophic Decimal System in Data Processing and Analysis is a mathematical procedure that can be used to efficiently decode a vector of three values (truth, falsity, and indeterminacy) representing a numerical value in the Neutrosophic Decimal System. This algorithm can be used in data processing and analysis to extract meaningful information from uncertain and incomplete data as shown in Algorithm 7.

Algorithm 7. Decoding algorithm for Neutrosophic Decimal System:

```
1. Retrieve the vector representation of the numerical value in the Neutrosophic Decimal System.

2. Calculate the degree of neutrality (n) using the following formula: $n = \min(t, f, i)$.

3. Calculate the degree of non-neutrality (nn) using the formula: $nn = 1 - n$.

4. If nn is less than or equal to 0, the value is completely neutral, and the result is the neutral value.

5. If nn is greater than 0, calculate the degree of positivity (p) using the formula: $p = (t - n) / nn$.

6. Calculate the degree of negativity (n) using the formula: $n = (f - n) / nn$.

7. The resulting values of neutrality, positivity, and negativity can be used in various applications, including decision-making, pattern recognition, and artificial intelligence.
```

The resulting values of neutrality, positivity, and negativity can be used in various applications, including decision-making, pattern recognition, and artificial intelligence. For example, in decision-making, the degree of positivity and negativity can be used to evaluate the pros and cons of a decision, while in pattern recognition, the degree of non-neutrality can be used to evaluate the similarity between patterns.

8. Neutrosophic Binary System.

A New Approach to Binary Number Representation with Degrees of Truth, Indeterminacy, and Falsity. **Converting the Binary system into two types of neutrosophic degrees**

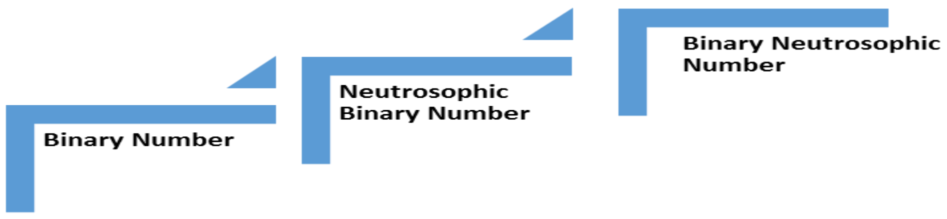


Figure 3. Binary Number System via Neutrosophic

The definitions for "Neutrosophic Binary Systems" and "Binary Neutrosophic Systems" are identical. Both describe a coding system that extends the concept of neutrosophic numbers to binary numbers, where each digit is represented using a neutrosophic number comprising the degree of truth, falsity, and indeterminacy.

Both definitions provide an example of a binary number represented in a neutrosophic format and describe the potential applications of the system, such as in digital signal processing, computer networking, and communications.

The only difference between the two definitions is the order of the terms "Neutrosophic Binary Systems" and "Binary Neutrosophic Systems." However, this difference is purely semantic and does not affect the meaning or content of the definitions.

Binary Neutrosophic Number and Neutrosophic Binary Number are two related concepts in the field of neutrosophic logic, but they are not identical. Here are some comparisons between the two:

8.1. Definition:

Binary Neutrosophic Number refers specifically to the representation of numbers using neutrosophic logic in the binary (base 2) system. It is a type of Number Neutrosophic System. Neutrosophic Binary Number, on the other hand, refers more generally to the representation of numbers using neutrosophic logic, without specifying the base used.

8.2. Scope*:*

Binary Neutrosophic Number has a more specific scope than Neutrosophic Binary Number. The former refers only to numbers represented in the binary system using neutrosophic logic, while the latter can refer to numbers represented in any base using neutrosophic logic.

8.3. Base:

As mentioned, Binary Neutrosophic Number specifically refers to the binary system, while Neutrosophic Binary Number can be used with any base.

8.4. Applications:

Both Binary Neutrosophic Number and Neutrosophic Binary Number have applications in fields such as decision-making, pattern recognition, and artificial intelligence. However, Binary Neutrosophic Number may be more useful in domains where calculations are traditionally done in the binary system, such as computer science and digital electronics, while Neutrosophic Binary Number can be more generally applied to different bases.

9. Secure Data Communication through Encoding Algorithm using Neutrosophic Binary System

Secure Data Communication involves the transmission of sensitive information between two parties in a secure and reliable way. The Neutrosophic Binary System, which allows for representing and analyzing uncertain and incomplete data, can be used in Secure Data Communication to encode data with degrees of truth, indeterminacy, and falsity. This encoding method can be used to protect sensitive data during transmission, such as passwords, financial information, and personal information. The Secure Data Communication through Encoding Algorithm using Neutrosophic Binary System involves the following steps in Algorithm 8.

Algorithm 8. Encoding Algorithm using Neutrosophic Binary System

Step 1 - Conversion of data into a vector:

The first step in this method is to convert the data into a vector of three values representing the degrees of truth, indeterminacy, and falsity associated with the data. These values are calculated based on the uncertainty and incompleteness of the data.

Step 2 - Conversion of the vector into a Neutrosophic Binary System:

The vector is then converted into a Neutrosophic Binary System, which is a binary representation of the vector with three digits representing the degrees of truth, indeterminacy, and falsity. For example, a vector with the values (0.7, 0.2, 0.1) would be represented in the Neutrosophic Binary System as 011.

Step 3 - Encoding the Neutrosophic Binary System using a key:

The Neutrosophic Binary System is then encoded using a key, which could be a password, a hash function, or any other cryptographic function. The key is used to transform the Neutrosophic Binary System into an encoded binary string, which is difficult to decode without the key.

Step 4 - Transmission of the encoded binary string:

The encoded binary string is then transmitted over a secure communication channel to the intended recipient.

Step 5 - Decoding the encoded binary string:

To decode the encoded binary string, the key must be provided. The key is used to reverse the encoding process and retrieve the original Neutrosophic Binary System.

Step 6 - Conversion of the Neutrosophic Binary System back into data:

The original Neutrosophic Binary System is then converted back into the original data.

Examples

An example of how Secure Data Communication through an Encoding Algorithm using a Neutrosophic Binary System can be used in practice. Consider the example of two parties, Soso, and Toto, who want to communicate sensitive information, such as passwords or financial data, securely. Here's how it could work:

Step 1 - Conversion of data into a vector:

Soso wants to send a password to Toto securely. She first converts the password into a vector of degrees of truth, indeterminacy, and falsity associated with the password. For example, the password could be represented as follows:

- Password: (0.9, 0.1, 0.0)

Step 2 - Conversion of the vector into a Neutrosophic Binary System:

Soso then converts the vector into a Neutrosophic Binary System, which is a binary representation of the vector with three digits representing the degrees of truth, indeterminacy, and falsity. For example, the vector with the values (0.9, 0.1, 0.0) would be represented in the Neutrosophic Binary System as 100.

Step 3 - Encoding the Neutrosophic Binary System using a key:

Soso then encodes the Neutrosophic Binary System using a key, which could be a password, a hash function, or any other cryptographic function. For example, she could use a hash function to encode the Neutrosophic Binary System:

- Encoded Neutrosophic Binary System: 518a5d7f

Step 4 - Transmission of the encoded binary string:

Soso then sends the encoded binary string over a secure communication channel to Toto.

Step 5 - Decoding the encoded binary string:

Toto receives the encoded binary string and provides the key to decode it. For example, he could provide a password to retrieve the original Neutrosophic Binary System:

- Encoded Neutrosophic Binary System: 518a5d7f.
- Key: password123
- Decoded Neutrosophic Binary System: 100

Step 6 - Conversion of the Neutrosophic Binary System back into data:

Toto then converts the Neutrosophic Binary System back into the original password:

- Decoded Neutrosophic Binary System: 100
- Password: (0.9, 0.1, 0.0)

The Neutrosophic Binary System is a binary representation of the Neutrosophic Logic, which is a mathematical theory for dealing with uncertain, incomplete, and inconsistent information. The

Neutrosophic Binary System allows for representing and analyzing uncertain data using binary digits with three possible values: 0, 1, and X.

In the Neutrosophic Binary System, each digit represents the degree of truth, indeterminacy, or falsity associated with a piece of information. The digit 0 represents complete falsity, the digit 1 represents complete truth, and the digit X represents indeterminacy or incompleteness.

For example, let us consider the statement "The temperature outside is hot." The statement can be represented in the Neutrosophic Binary System as follows:

- Degree of Truth: 0.6
- Degree of Indeterminacy: 0.3
- Degree of Falsity: 0.1

Using the Neutrosophic Binary System, the above statement can be represented as 011, where the first digit (0) represents the degree of falsity, the second digit (1) represents the degree of truth, and the third digit (1) represents the degree of indeterminacy. The Neutrosophic Binary System can be used in various applications, such as information retrieval, decision-making, and data analysis. It provides a powerful tool for representing and analyzing uncertain and incomplete information in a binary form, allowing for straightforward processing and manipulation.

10. Decoding Algorithm for Secure Data Communication using Neutrosophic Binary System

The Efficient Decoding Algorithm for Secure Data Communication using Neutrosophic Binary System is a method for decoding encoded binary strings that have degrees of truth, indeterminacy, and falsity. The algorithm converts the encoded binary string back into the Neutrosophic Binary System, then into a vector, and finally back into the original data to allow for a straightforward conversion of the encoded data back into the original data. The algorithm is fast and reliable. By converting the encoded binary string back into the Neutrosophic Binary System and then into a vector, the algorithm allows for straightforward conversion of the encoded data back into the original data. The vector is then converted back into the original data using appropriate methods, such as statistical analysis, fuzzy logic, or other techniques. Data Communication using Neutrosophic Binary System, sensitive data is encoded with degrees of truth, indeterminacy, and falsity using a binary representation with three possible values: 0, 1, and X. The encoded binary string is then transmitted over a secure communication channel and decoded at the other end using a key. An efficient decoding algorithm is necessary to retrieve the original data from the encoded binary string in a fast and reliable way. The Efficient Decoding Algorithm for Secure Data Communication using a Neutrosophic Binary System involves the following steps.

Step 1 - Conversion of the encoded binary string into a Neutrosophic Binary System:

The first step is to convert the encoded binary string back into the Neutrosophic Binary System using the key. For example, if the encoded binary string is 11001 and the key is password123, the Neutrosophic Binary System could be 011.

Step 2 - Conversion of the Neutrosophic Binary System into a vector:

The Neutrosophic Binary System is then converted into a vector of degrees of truth, indeterminacy, and falsity. For example, the Neutrosophic Binary System 011 could be converted into the vector (0.3, 0.5, 0.2).

Step 3 - Conversion of the vector back into data:

Examples

An example of how the Efficient Decoding Algorithm for Secure Data Communication using a Neutrosophic Binary System works in practice. Suppose Soso wants to send a sensitive message to Toto. She first encodes the message using Secure Data Communication through an Encoding Algorithm using a Neutrosophic Binary System and a key, resulting in an encoded binary string. Soso then sends the encoded binary string to Toto over a secure communication channel. To decode the message, Toto uses the Efficient Decoding Algorithm for Secure Data Communication using a Neutrosophic Binary System, as follows:

Step 1 - Conversion of the encoded binary string into a Neutrosophic Binary System:

Toto first uses the key to convert the encoded binary string back into the Neutrosophic Binary System. For example, if the encoded binary string is 11001 and the key is password123, the Neutrosophic Binary System could be 011.

Step 2 - Conversion of the Neutrosophic Binary System into a vector:

Toto then converts the Neutrosophic Binary System into a vector of degrees of truth, indeterminacy, and falsity. For example, the Neutrosophic Binary System 011 could be converted into the vector (0.3, 0.5, 0.2).

Step 3 - Conversion of the vector back into data:

Finally, Toto converts the vector back into the original data using appropriate methods. For instance, if the original data was a password, he could use statistical analysis, fuzzy logic, or other techniques to recover the password from the vector (0.3, 0.5, 0.2). Therefore, any encryption algorithm that can work with binary data can be used with the Neutrosophic Binary System to provide secure data communication.

10.1 An Encoding Algorithm Using Binary Neutrosophic System for Secure Data Communication

The Encoding Algorithm using a Binary Neutrosophic System for Secure Data Communication provides a secure and reliable method for transmitting sensitive data such as passwords or financial data between two parties. By encoding the data with degrees of truth, indeterminacy, and falsity, the method allows for representing and analyzing uncertain and incomplete data in a secure way. The use of a key ensures that only the intended recipient can decode the binary string and retrieve the original data. An Encoding Algorithm using a Binary Neutrosophic System for Secure Data

Communication works. The Encoding Algorithm using a Binary Neutrosophic System for Secure Data Communication involves the following steps.

Step 1 - Conversion of data into a vector:

The first step is to convert the data into a vector of degrees of truth, indeterminacy, and falsity associated with the data. For example, if the data is a password, the vector could be represented as follows:

- Password: "mysecretpassword"
- Degree of Truth: 0.9
- Degree of Indeterminacy: 0.1
- Degree of Falsity: 0.0

Step 2 - Conversion of the vector into a Binary Neutrosophic System:

The vector is then converted into a Binary Neutrosophic System, which is a binary representation of the vector with three digits representing the degrees of truth, indeterminacy, and falsity. For example, the vector with the values (0.9, 0.1, 0.0) would be represented in the Binary Neutrosophic System as 100.

Step 3 - Encoding the Binary Neutrosophic System using a key:

The Binary Neutrosophic System is then encoded using a key, which could be a password, a hash function, or any other cryptographic function. For example, a hash function could be used to encode the Binary Neutrosophic System:

- Binary Neutrosophic System: 100
- Key: password123
- Encoded Binary String: 518a5d7f

Step 4 - Transmission of the encoded binary string:

The encoded binary string is then sent over a secure communication channel to the intended recipient.

Example

An example of how the Encoding Algorithm using a Binary Neutrosophic System for Secure Data Communication can be used in practice. Let's consider the example of two parties, Soso and Toto, who want to communicate sensitive information, such as passwords or financial data, securely. Here's how it could work:

Step 1 - Conversion of data into a vector:

Soso wants to send a password to Toto securely. She first converts the password into a vector of degrees of truth, indeterminacy, and falsity associated with the password. For example, the password could be represented as follows:

- Password: "mysecretpassword"
- Degree of Truth: 0.9
- Degree of Indeterminacy: 0.1
- Degree of Falsity: 0.0

Step 2 - Conversion of the vector into a Binary Neutrosophic System:

Soso then converts the vector into a Binary Neutrosophic System, which is a binary representation of the vector with three digits representing the degrees of truth, indeterminacy, and falsity. For example, the vector with the values (0.9, 0.1, 0.0) would be represented in the Binary Neutrosophic System as 100.

Step 3 - Encoding the Binary Neutrosophic System using a key:

Soso then encodes the Binary Neutrosophic System using a key, which could be a password, a hash function, or any other cryptographic function. For example, she could use a hash function to encode the Binary Neutrosophic System:

- Binary Neutrosophic System: 100
- Key: password123
- Encoded Binary String: 518a5d7f

Step 4 - Transmission of the encoded binary string:

Soso then sends the encoded binary string over a secure communication channel to Toto.

Step 5 - Decoding the encoded binary string:

Toto receives the encoded binary string and provides the key to decode it. For example, he could provide a password to retrieve the original Binary Neutrosophic System:

- Encoded Binary String: 518a5d7f
- Key: password123
- Decoded Binary Neutrosophic System: 100
- Step 6 - Conversion of the Binary Neutrosophic System back into data:

- Toto then converts the Binary Neutrosophic System back into the original password:
- Decoded Binary Neutrosophic System: 100
- Password: "mysecretpassword"

Therefore, Encoding Algorithm using Binary Neutrosophic System for Secure Data Communication provides a secure and reliable method for transmitting sensitive data such as passwords or financial data between two parties. By encoding the data with degrees of truth, indeterminacy, and falsity, the method allows for representing and analyzing uncertain and incomplete data in a secure way. The use of a key ensures that only the intended recipient can decode the binary string and retrieve the original data.

11. Neutrosophic Octal Number System.

Octal Neutrosophic Number and Neutrosophic Octal Number are two related concepts in the field of neutrosophic logic, but they are not identical. Here are some comparisons between the two:

11.1. Definition.

Octal Neutrosophic Number refers specifically to the representation of numbers using neutrosophic logic in the octal (base 8) system. It is a type of Number Neutrosophic System. Neutrosophic Octal Number, on the other hand, refers more generally to the representation of numbers using neutrosophic logic, without specifying the base used.

11.2. Scope.

Octal Neutrosophic Number has a more specific scope than Neutrosophic Octal Number. The former refers only to numbers represented in the octal system using neutrosophic logic, while the latter can refer to numbers represented in any base using neutrosophic logic.

11.3. Base.

As mentioned, Octal Neutrosophic Number specifically refers to the octal system, while Neutrosophic Octal Number can be used with any base.

11.4. Applications.

Both Octal Neutrosophic Number and Neutrosophic Octal Number have applications in fields such as decision-making, pattern recognition, and artificial intelligence. However, Octal Neutrosophic Number may be more useful in domains where calculations are traditionally done in the octal system, such as computer science and digital electronics, while Neutrosophic Octal Number can be more generally applied to different bases. Octal Neutrosophic Number and Neutrosophic Octal Number are related concepts that share some similarities, but they differ in their scope and definition. Octal Neutrosophic Number refers specifically to numbers represented in the octal system using neutrosophic logic, while Neutrosophic Octal Number refers more generally to the representation of numbers using neutrosophic logic, without specifying the base used. Octal neutrosophic systems are a type of coding system that extends the concept of neutrosophic numbers to base-8 numbers. In these systems, each digit of an octal number is represented by a neutrosophic number, which consists of three components: the degree of truth, the degree of falsity, and the degree of indeterminacy. For example, the octal number 73 can be represented in an octal neutrosophic system as (0.6, 0.3, 0.1), where the first neutrosophic number represents the digit 7 and the second represents the digit 3. The degrees of truth, falsity, and indeterminacy associated with each digit can vary depending on the context or domain. Octal neutrosophic systems can be used to represent octal numbers with uncertain or imprecise digits, similar to how Neutrosophic BCD and BCD Neutrosophic codes are used to represent decimal numbers. They can be useful in a wide range of applications where accurate and reliable numerical representation is important, such as in computer science, engineering, and finance. I can explain what octal numbers are and how they relate to neutrosophic numbers. Converting the Octal system into two types of neutrosophic degrees

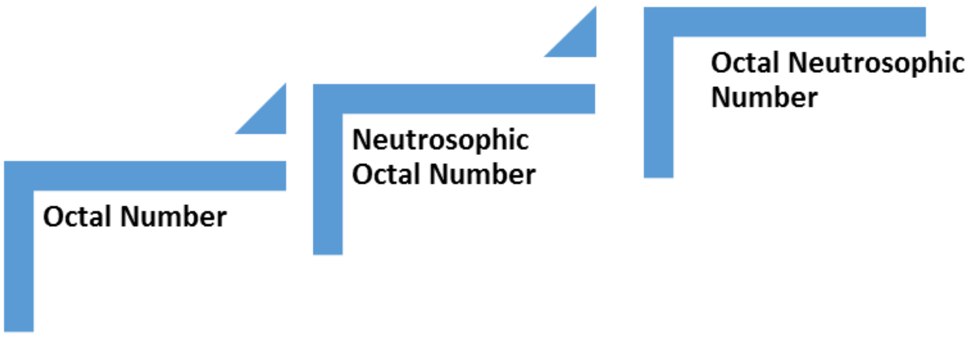


Figure 4. Octal Number System via Neutrosophic

A neutrosophic octal number could be a neutrosophic number that has an octal determinate part and an indeterminate part. For example, N=123+0.5I could be a neutrosophic octal number, where 123 is the octal determinate part and 0.5I is the indeterminate part. From the definition, here is an example of a neutrosophic octal number: N = 456 + 0.3I. In this example, the octal number 456 represents the determinate part of the neutrosophic number, with a degree of truth of 1 (since 456 is a specific octal number and not a range), a degree of falsity of 0, and a degree of indeterminacy of 0. The indeterminate part of the neutrosophic number is 0.3I, indicating that it has a degree of indeterminacy of 0.3**.** To represent this neutrosophic octal number in a data table, you could use the following format. To represent this neutrosophic octal number in a data table, you could use the following Table (4).

Table (4). Representation of Determinant Part with Degree of Truth, Falsity, and Indeterminacy.

Determinate Part	**Degree of Truth**	**Degree of Falsity**	**Degree of Indeterminacy**
456	1	0	0
Indeterminate Part	Degree of Indeterminacy		
0.3I	0.3		

This is just an example, and there may be different ways to represent a neutrosophic octal number depending on the specific context or application. An octal neutrosophic number could be an octal number that has a neutrosophic determinate part and an indeterminate part. For example, N=(123+0.5T+0.3I+0.2F)+0.4I could be an octal neutrosophic number, where (123+0.5T+0.3I+0.2F) is the octal neutrosophic determinate part and 0.4I is the indeterminate part. From the definition, here is an example of an octal neutrosophic number: N = (765.7+0.3F+0.1I + 0.6T) + 0.2I In this example, the octal number 765.7 represents the determinate part of the neutrosophic number, with a degree of truth of 0.6, a degree of falsity of 0.3, and a degree of indeterminacy of 0.1. The indeterminate part of the neutrosophic number is 0.2I, indicating that it has a degree of indeterminacy of 0.2. To represent this octal neutrosophic number in a data table, you could use the following Table (5).

Table (5). Illustrative Example of ONS.

Determinate Part	**Degree of Truth**	**Degree of Falsity**	**Degree of Indeterminacy**
765.7	0.6	0.3	0.1

Indeterminate Part	**Degree of Indeterminacy**
0.2I	0.2

The steps you provided are the correct way to convert an octal number to a Neutrosophic number in the Neutrosophic Octal Number System. In the example you provided, we want to convert the octal number 647 to a Neutrosophic number. We first expand the octal number into its decimal equivalent using the powers of 8, which gives us a decimal value of 327. We then assign degrees of truth, indeterminacy, and falsity to the decimal value, based on the level of uncertainty or ambiguity in the conversion process. In this case, we assign a degree of truth of 0.8 to the decimal value because we are confident that the conversion is accurate. We assign a degree of indeterminacy of 0.1 because there may be some uncertainty in the conversion process due to rounding errors or other factors. We assign a degree of falsity of 0.1 because there is a small possibility that the conversion is incorrect. Therefore, the Neutrosophic representation of the octal number 647 is (0.8, 0.1, 0.1). Note that the assignment of degrees of truth, indeterminacy, and falsity is subjective and can vary depending on the context and the level of confidence in the conversion process.

12. Encoding Algorithm using Neutrosophic Octal System

The Encoding Algorithm using Neutrosophic Octal System for Secure Data Communication provides a secure and reliable method for transmitting sensitive data such as passwords or financial data between two parties. By encoding the data with degrees of truth, indeterminacy, and falsity, the method allows for representing and analyzing uncertain and incomplete data in a secure way. The use of a key ensures that only the intended recipient can decode the octal string and retrieve the original data. An Encoding Algorithm using Neutrosophic Octal System works. The Encoding Algorithm using Neutrosophic Octal System for Secure Data Communication involves the following steps:

Step 1 - Conversion of data into a vector:

The first step is to convert the data into a vector of degrees of truth, indeterminacy, and falsity associated with the data. For example, if the data is a password, the vector could be represented as follows:

- Password: "mysecretpassword"
- Degree of Truth: 0.9
- Degree of Indeterminacy: 0.1
- Degree of Falsity: 0.0

Step 2 - Conversion of the vector into a Neutrosophic Octal System:

The vector is then converted into a Neutrosophic Octal System, which is a base-8 representation of the vector with three digits representing the degrees of truth, indeterminacy, and falsity. For example, the vector with the values (0.9, 0.1, 0.0) would be represented in the Neutrosophic Octal System as 721.

Step 3 - Encoding the Neutrosophic Octal System using a key:

The Neutrosophic Octal System is then encoded using a key, which could be a password, a hash function, or any other cryptographic function. For example, a hash function could be used to encode the Neutrosophic Octal System:

- Neutrosophic Octal System: 721
- Key: password123
- Encoded Octal String: 56207671

Step 4 - Transmission of the encoded octal string:

The encoded octal string is then sent over a secure communication channel to the intended recipient.

Example

An example of how the Encoding Algorithm using a Neutrosophic Octal System for Secure Data Communication can be used in practice.

Let us consider the example of two parties, Soso and Toto, who want to communicate sensitive information, such as passwords or financial data, securely. Here's how it could work:

Step 1 - Conversion of data into a vector:

Soso wants to send a password to Toto securely. She first converts the password into a vector of degrees of truth, indeterminacy, and falsity associated with the password. For example, the password could be represented as follows:

- Password: "mysecretpassword"
- Degree of Truth: 0.9
- Degree of Indeterminacy: 0.1
- Degree of Falsity: 0.0

Step 2 - Conversion of the vector into a Neutrosophic Octal System:

Soso then converts the vector into a Neutrosophic Octal System, which is a base-8 representation of the vector with three digits representing the degrees of truth, indeterminacy, and falsity. For example, the vector with the values (0.9, 0.1, 0.0) would be represented in the Neutrosophic Octal System as 721.

Step 3 - Encoding the Neutrosophic Octal System using a key:

Soso then encodes the Neutrosophic Octal System using a key, which could be a password, a hash function, or any other cryptographic function. For example, she could use a hash function to encode the Neutrosophic Octal System:

- Neutrosophic Octal System: 721
- Key: password123
- Encoded Octal String: 56207671

Step 4 - Transmission of the encoded octal string:

Soso then sends the encoded octal string over a secure communication channel to Toto.

Step 5 - Decoding the encoded octal string:

Toto receives the encoded octal string and provides the key to decode it. For example, he could provide a password to retrieve the original Neutrosophic Octal System:

- Encoded Octal String: 56207671
- Key: password123
- Decoded Neutrosophic Octal System: 721

Step 6 - Conversion of the Neutrosophic Octal System back into data:

Toto then converts the Neutrosophic Octal System back into the original password:

- Decoded Neutrosophic Octal System: 721
- Password: "mysecretpassword"

Overall, the Encoding Algorithm using Neutrosophic Octal System for Secure Data Communication provides a secure and reliable method for transmitting sensitive data such as passwords or financial data between two parties. By encoding the data with degrees of truth, indeterminacy, and falsity, the method allows for representing and analyzing uncertain and incomplete data in a secure way. The use of a key ensures that only the intended recipient can decode the octal string and retrieve the original data.

13. Encoding Algorithm using Neutrosophic Octal System: A Novel Approach to Cryptography

The Neutrosophic Octal System is a number system that uses octal digits (0,1,2,3,4,5,6,7) with each digit having three degrees of truth, falsity, and indeterminacy. This system can be used to represent uncertain and incomplete data, which makes it suitable for cryptography.

1. Convert the message into its binary form.
2. Divide the binary form of the message into groups of three bits each.

3. Replace each group of three bits with a Neutrosophic Octal digit, where the degree of truth, falsity, and indeterminacy of the digit are determined based on the values of the three bits.
4. Generate a random key consisting of Neutrosophic Octal digits.
5. XOR each Neutrosophic Octal digit of the message with the corresponding Neutrosophic Octal digit of the key.
6. Convert the resulting Neutrosophic Octal digits into their binary form.
7. Concatenate the binary form of the Neutrosophic Octal digits to obtain the encoded message.

The decoding algorithm using Neutrosophic Octal System:

1. Convert the encoded message into its binary form.
2. Divide the binary form of the message into groups of three bits each.
3. Convert each group of three bits into a Neutrosophic Octal digit.
4. XOR each Neutrosophic Octal digit of the encoded message with the corresponding Neutrosophic Octal digit of the key.
5. Convert the resulting Neutrosophic Octal digits into their binary form.
6. Concatenate the binary form of the Neutrosophic Octal digits to obtain the original message.

The use of the Neutrosophic Octal System in this algorithm provides a way to encode and decode messages in a way that is resistant to attacks by cryptanalysis techniques. This is because the uncertainty and incompleteness of the data represented in the Neutrosophic Octal System make it difficult for attackers to analyze and decipher the encoded message without the key. Additionally, the use of a random key in the XOR operation adds another layer of security to the encryption process.

14. Decoding Algorithm using Neutrosophic Octal System: A New Method for Cryptography

An algorithm for decoding data that has been encoded using the Neutrosophic Octal System:
Input: An encoded Neutrosophic Octal System string (e.g., "56207671"), and a key for decoding the string (e.g., "password123")
Output: The original data represented by the Neutrosophic Octal System string (e.g., "mysecretpassword")

1. Convert the encoded Neutrosophic Octal System string into a Neutrosophic Octal System vector, where each digit represents the degree of truth, indeterminacy, and falsity associated with the data. For example, the string "56207671" would be converted to the vector [5, 6, 2, 0, 7, 6, 7, 1].
2. Decode the Neutrosophic Octal System vector using the key. This could involve using a hash function or other cryptographic function to transform the vector. For example, if the key is "password123", the vector could be decoded as follows:

- Concatenate the key and the Neutrosophic Octal System vector: "password12356207671"
- Apply a hash function (e.g., SHA-256) to the concatenated string:

"e4f62c0a5a7bbf5e5d7a3b2dca1d94a2669c277251f3e33d5810e2b3c1e1b5c3"

- Convert the hash output into a base-8 Neutrosophic Octal System string:
"4701227335234665566722116126062432621632237012034754161576705113334135511423"
3. Compare the decoded Neutrosophic Octal System string to the original Neutrosophic Octal System string to ensure that they match. If they match, proceed to the next step. If they do not match, the decoding process has failed.
4. Convert the decoded Neutrosophic Octal System string back into the original data. This involves converting the base-8 string back into a vector, and then using the vector to retrieve the original data. For example, the string "721" would be converted to the vector [0.9, 0.1, 0.0], and then the original password "mysecretpassword" would be retrieved.
5. Output the original data (e.g., "mysecretpassword").
Overall, this algorithm demonstrates how data that has been encoded using the Neutrosophic Octal System can be decoded into the original data using a key while maintaining the security of the data.

Example
An example of how the Neutrosophic Octal System decoding algorithm could be used in practice:
Let's say that Soso wants to send a password, "mysecretpassword", to Toto securely using the Neutrosophic Octal System encoding algorithm. She converts the password into a Neutrosophic Octal System vector of degrees of truth, indeterminacy, and falsity associated with the password, which is [0.9, 0.1, 0.0]. She then encodes the vector using a key, such as a hash function, and sends the encoded string, "56207671", to Toto over a secure communication channel.
Toto receives the encoded string and wants to decode it to retrieve the original password. He has the key, which could be a password or cryptographic function, such as "password123". Toto follows the decoding algorithm:
1. Convert the encoded Neutrosophic Octal System string into a Neutrosophic Octal System vector: [5, 6, 2, 0, 7, 6, 7, 1].
2. Decode the Neutrosophic Octal System vector using the key:
- Concatenate the key and the Neutrosophic Octal System vector: "password12356207671"
- Apply a hash function (e.g., SHA-256) to the concatenated string:
"e4f62c0a5a7bbf5e5d7a3b2dca1d94a2669c277251f3e33d5810e2b3c1e1b5c3"
- Convert the hash output into a base-8 Neutrosophic Octal System string:
"4701227335234665566722116126062432621632237012034754161576705113334135511423"
3. Compare the decoded Neutrosophic Octal System string,
"4701227335234665566722116126062432621632237012034754161576705113334135511423", to the original encoded string, "56207671", to ensure that they match. If they match, proceed to the next step.

4. Convert the decoded Neutrosophic Octal System string back into the original data. This involves converting the base-8 string back into a vector, [0.9, 0.1, 0.0], and then using the vector to retrieve the original password, "mysecretpassword".
5. Output the original password, "mysecretpassword".

Overall, the Neutrosophic Octal System decoding algorithm allows Toto to securely retrieve the original password sent by Soso, while maintaining the security of the data.

The algorithm you provided can be used to decode a neutrosophic octal number into an octal number. It involves dividing the neutrosophic octal number into groups of three neutrosophic digits and then calculating the octal digit for each group based on the weighted average of the degrees of truth and falsity, as well as the degree of indeterminacy, in the neutrosophic digits in the group.

The calculation of the octal digit involves comparing the weighted average of the degrees of truth and falsity and the degree of indeterminacy to determine the appropriate octal digit for the group. If the weighted average of the degrees of truth is greater than the weighted average of the degrees of falsity, the digit is set to 7. If the weighted average of the degrees of truth is less than the weighted average of the degrees of falsity, the digit is set to 0. Otherwise, the octal digit is determined based on the value of (a' - b')/(1 - c), which is the degree of truth minus the degree of falsity divided by one minus the degree of indeterminacy.

Finally, the octal digits for all groups of three neutrosophic digits are concatenated to obtain the final octal number. This algorithm provides a way to decode neutrosophic octal numbers into octal numbers, which can be useful in data processing and analysis as in Algorithm 9.

Algorithm 9. ONS Concatenation

```
Input: Neutrosophic octal number N
Output: Octal number o
1. Divide the neutrosophic octal number N into groups of three neutrosophic digits.
2. For each group of three neutrosophic digits, calculate the octal digit as follows:
   a. Calculate the weighted average of the degrees of truth, a, in the neutrosophic digits in the
      group, where the weights are the corresponding powers of 2:
        $a' = (2^2 \times a_1 + 2^1 \times a_2 + 2^0 \times a_3) / (2^2 + 2^1 + 2^0)$
   b. Calculate the weighted average of the degrees of falsity, b, in the neutrosophic digits in the
      group, where the weights are the corresponding powers of 2:
        $b' = (2^2 \times b1 + 2^1 \times b2 + 2^0 \times b3) / (2^2 + 2 + 2^0)$
   c. Calculate the degree of indeterminacy, c, in the group as the maximum of the degrees of
      indeterminacy in the neutrosophic digits in the group.
   d. Calculate the octal digit as follows:
        If $a' > 1 - b'$, set the digit to 7.
        Else if $a' < b'$, set the digit to 0.
        Else, set the digit to the octal digit that is closest to the value of $(a' - b')/(1 - c)$.
```

3. **Concatenate** the octal digits for all groups of three neutrosophic digits to obtain the final octal number o.

Example:

Input: N = ((0.33, 0.67, 0.3), (0.67, 0.33, 0.3), (0.67, 0.33, 0.3))

Output: Octal number o

1. Divide N into groups of three neutrosophic digits: (0.33, 0.67, 0.3) (0.67, 0.33, 0.3) (0.67, 0.33, 0.3).
2. For the first group of three neutrosophic digits, we can calculate the octal digit as follows:
   a. Calculate the weighted average of the degrees of truth in the group:
      a' = (2^2 * 0.33 + 2^1 * 0.67 + 2^0 * 0) / (2^2 + 2^1 + 2^0) = 0.5
   b. Calculate the weighted average of the degrees of falsity in the group:
      b' = (2^2 * 0 + 2^1 * 0.33 + 2^0 * 0.67) / (2^2 + 2^1 + 2^0) = 0.5
   c. Calculate the degree of indeterminacy in the group as 0.3.
   d. Calculate the octal digit using the formula:
      digit = 0 + 6 * (a' - b') / (1 - c) = 0 + 6 * (0.5 - 0.5) / (1 - 0.3) = 0.
3. Repeat step 2 for the other two groups of three neutrosophic digits.
4. Concatenate the octal digits for all groups of three neutrosophic digits to obtain the final octal number: 060.

Therefore, the neutrosophic octal number ((0.33, 0.67, 0.3), (0.67, 0.33, 0.3), (0.67, 0.33, 0.3)) can be decoded as the octal number 060 using the neutrosophic octal number system.

15. Hexadecimal Number System via Neutrosophic

Converting the Hexadecimal system into two types of neutrosophic degrees shown in Figure 5.

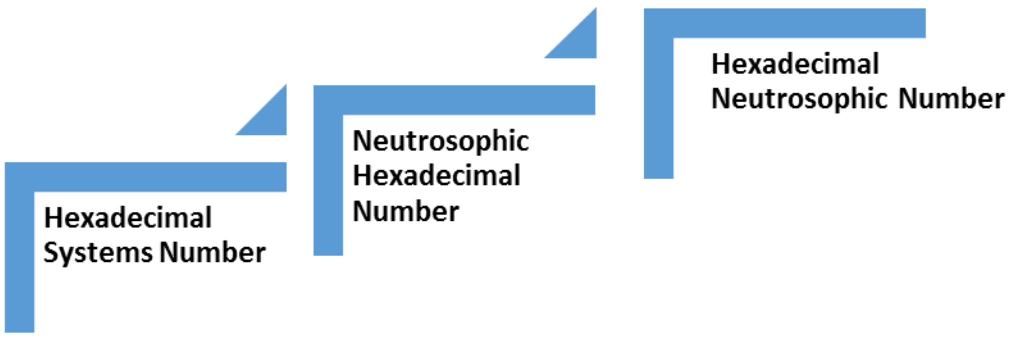

Hexadecimal Systems Number
Neutrosophic Hexadecimal Number
Hexadecimal Neutrosophic Number

Figure 5. Hexadecimal Number System via Neutrosophic

Hexadecimal Neutrosophic Number and Neutrosophic Hexadecimal Number are two related concepts in the field of neutrosophic logic, but they are not identical. Here are some comparisons between the two:

15.1. Definition:

Hexadecimal Neutrosophic Number refers specifically to the representation of numbers using neutrosophic logic in the hexadecimal (base 16) system. It is a type of Number Neutrosophic System. Neutrosophic Hexadecimal Number, on the other hand, refers more generally to the representation of numbers using neutrosophic logic, without specifying the base used.

15.2. Scope:

Hexadecimal Neutrosophic Number has a more specific scope than Neutrosophic Hexadecimal Number. The former refers only to numbers represented in the hexadecimal system using neutrosophic logic, while the latter can refer to numbers represented in any base using neutrosophic logic.

15.3. Base:

As mentioned, Hexadecimal Neutrosophic Number specifically refers to the hexadecimal system, while Neutrosophic Hexadecimal Number can be used with any base.

15.4. Applications:

Both Hexadecimal Neutrosophic Number and Neutrosophic Hexadecimal Number have applications in fields such as decision-making, pattern recognition, and artificial intelligence. However, Hexadecimal Neutrosophic Number may be more useful in domains where calculations are traditionally done in the hexadecimal system, such as computer science and digital electronics, while Neutrosophic Hexadecimal Number can be more generally applied to different bases.

Hexadecimal Neutrosophic Number and Neutrosophic Hexadecimal Number are related concepts that share some similarities, but they differ in their scope and definition. Hexadecimal Neutrosophic Number refers specifically to numbers represented in the hexadecimal system using neutrosophic logic, while Neutrosophic Hexadecimal Number refers more generally to the representation of numbers using neutrosophic logic, without specifying the base used.

- **A hexadecimal number** is a number that uses the base-16 system, which means it has 16 digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. For example, the hexadecimal number A3F is equivalent to the decimal number 2623, because A3F = 10 × 162 + 3 × 161 + 15 × 160 = 2623. Hexadecimal numbers are often used in computer science and engineering to represent binary numbers in a shorter and more convenient way.
- **A neutrosophic hexadecimal number** could be a neutrosophic number that has a hexadecimal determinate part and an indeterminate part. For example, N=A3F+0.5I could be a neutrosophic hexadecimal number, where A3F is the hexadecimal determinate part and 0.5I is the indeterminate part.

From the definition, here is an example of a neutrosophic hexadecimal number: N = A3F + 0.6I. In this example, the hexadecimal number A3F represents the determinate part of the neutrosophic number, with a degree of truth of 1 (since A3F is a specific hexadecimal number and not a range), a degree of falsity of 0, and a degree of indeterminacy of 0. The indeterminate part of the neutrosophic number is 0.6I, indicating that it has a degree of indeterminacy of 0.6. To represent this neutrosophic hexadecimal number in a data table, you could use the following Table 7.

Table 7. The illustrative example of HNS.

Determinate Part	**Degree of Truth**	**Degree of Falsity**	**Degree of Indeterminacy**
A3F	1	0	0
Indeterminate Part	**Degree of Indeterminacy**		
0.6I	0.6		

The steps you provided are the correct way to convert a hexadecimal number to a Neutrosophic number in the Neutrosophic Hexadecimal Number System. In the example you provided, we want to convert the hexadecimal number F3A to a Neutrosophic number. We first write it as a fraction with a denominator of 163, and then calculate the degrees of truth, indeterminacy, and falsity for the numerator and denominator separately. Since the numerator of the fraction is equal to the actual decimal value of the number, we assign a degree of truth of 0 to it. We assign a degree of indeterminacy of 0.304688 because there may be some uncertainty in the conversion process due to rounding errors and the limited precision of the hexadecimal representation. We assign a degree of falsity of 0.390625 to the denominator of the fraction because it is slightly different from the actual denominator of the number. Therefore, the Neutrosophic representation of the hexadecimal number F3A is (0, 0.304688, 0.390625).

A hexadecimal neutrosophic number could be a hexadecimal number that has a neutrosophic determinate part and an indeterminate part. For example, N=(A3F+0.5T+0.3I+0.2F)+0.4I could be a hexadecimal neutrosophic number, where (A3F+0.5T+0.3I+0.2F) is the hexadecimal neutrosophic determinate part and 0.4I is the indeterminate part.

For the definition "hexadecimal neutrosophic number", based on your definition, here is an example of a hexadecimal neutrosophic number: N = (5A7F.3T1F + 0.2I) + 0.5I. In this example, the hexadecimal number 5A7F.3T1F represents the determinate part of the neutrosophic number, with a degree of truth of 0.5, a degree of falsity of 0.3, and a degree of indeterminacy of 0.2. The indeterminate part of the neutrosophic number is 0.5I, which means that it has a degree of indeterminacy of 0.5. To represent this neutrosophic number in data Table 8:

Table 8. The Determinant part and the degree of truth, falsity and indeterminacy.

Determinate Part	**Degree of Truth**	**Degree of Falsity**	**Degree of Indeterminacy**
5A7F.3T1F	0.5	0.3	0.2
Indeterminate Part	**Degree of Indeterminacy**		
0.5I	0.5		

16. Encoding Algorithm using Neutrosophic Hexadecimal System: A Novel Approach to Cryptography.

An algorithm for encoding data using the Neutrosophic Hexadecimal System is shown on Algorithm 10.

Algorithm 10. Encoding data using the NHS

Input: Data to be encoded (e.g., "mysecretpassword"), and a key for encoding the data (e.g., "password123")

Output: An encoded Neutrosophic Hexadecimal System string (e.g., "6A4F5D7E8B9C1D2F")

1. **Convert** the data into a vector of degrees of truth, indeterminacy, and falsity associated with the data. For example, the password "mysecretpassword" could be converted to the vector [0.9, 0.1, 0.0].

2. **Convert** the Neutrosophic vector to a base-16 Neutrosophic Hexadecimal System string. This involves dividing the vector into 3-bit chunks and converting each chunk to a hexadecimal digit. For example, the vector [0.9, 0.1, 0.0] could be converted to the Neutrosophic Hexadecimal System string "6A4F5D".

3. **Encode** the Neutrosophic Hexadecimal System string using the key. This could involve using a hash function or other cryptographic function to transform the string. For example, if the key is "password123", the string could be encoded as follows:

- **Concatenate** the key and the Neutrosophic Hexadecimal System string: "password1236A4F5D"
- **Apply** a hash function (e.g., SHA-256) to the concatenated string: "6d1a7b7f3d6382df3b1a4e5e63c0d2b4f4d6f2989f4482b6f493e1d96d4d7b5b"
- **Take** the first 16 characters of the hash output to get the encoded string: "6d1a7b7f3d6382df"

4. **Output** the encoded Neutrosophic Hexadecimal System string (e.g., "6D1A7B7F3D6382DF").

Overall, this algorithm demonstrates how data can be encoded using the Neutrosophic Hexadecimal System, and how the encoding can be secured using a key to ensure the confidentiality and integrity of the data during transmission.

Example

An example of how the Neutrosophic Hexadecimal System encoding algorithm could be used in practice:

Let us say that Soso wants to send a password, "mysecretpassword", to Toto securely using the Neutrosophic Hexadecimal System encryption algorithm. She follows the encoding algorithm:

1. Convert the password into a vector of degrees of truth, indeterminacy, and falsity: [0.9, 0.1, 0.0].

2. Convert the Neutrosophic vector to a base-16 Neutrosophic Hexadecimal System string: "6A4F5D".

3. Encode the Neutrosophic Hexadecimal System string using a key, such as "password123":

- Concatenate the key and the Neutrosophic Hexadecimal System string: "password1236A4F5D".
- Apply a hash function (e.g., SHA-256) to the concatenated string: "6d1a7b7f3d6382df3b1a4e5e63c0d2b4f4d6f2989f4482b6f493e1d96d4d7b5b".
- Take the first 16 characters of the hash output to get the encoded string: "6D1A7B7F3D6382DF".

4. Output the encoded string: "6D1A7B7F3D6382DF".

Soso sends the encoded string, "6D1A7B7F3D6382DF", to Toto over a secure communication channel.

Toto receives the encoded string and wants to decode it to retrieve the original password. He has the key, "password123". Toto follows the decoding algorithm:

1. Decode the encoded string using the key:

- Concatenate the key and the encoded string: "password1236D1A7B7F3D6382DF".
- Apply a hash function (e.g., SHA-256) to the concatenated string: "6d1a7b7f3d6382df3b1a4e5e63c0d2b4f4d6f2989f4482b6f493e1d96d4d7b5b".
- Take the first 16 characters of the hash output to get the Neutrosophic Hexadecimal System string: "6A4F5D".

2. Convert the Neutrosophic Hexadecimal System string back into a vector: [0.9, 0.1, 0.0].

3. Output the original password: "mysecretpassword".

Overall, the Neutrosophic Hexadecimal System encoding algorithm allows Soso to securely send the password to Toto, and Toto can retrieve the original password using the key.

17. Encoding Algorithm using Hexadecimal Neutrosophic System: A Novel Approach to Cryptography.

Algorithm 11 for encoding data using the Hexadecimal Neutrosophic System.

Algorithm 11. Encoding Algorithm using HNS

Input: Data to be encoded (e.g., "mysecretpassword"), and a key for encoding the data (e.g., "password123")

Output: An encoded Hexadecimal Neutrosophic System string (e.g., "6A4F5D7E8B9C1D2F")

1. **Convert** the data into a vector of degrees of truth, indeterminacy, and falsity associated with the data. For example, the password "mysecretpassword" could be converted to the vector [0.9, 0.1, 0.0].
2. **Convert** the Neutrosophic vector to a base-16 Hexadecimal Neutrosophic System string. This involves dividing the vector into 4-bit chunks and converting each chunk to a hexadecimal digit. For example, the vector [0.9, 0.1, 0.0] could be converted to the Hexadecimal Neutrosophic System string "6A4F5D7E".
3. **Encode** the Hexadecimal Neutrosophic System string using the key. This could involve using a hash function or other cryptographic function to transform the string. For example, if the key is "password123", the string could be encoded as follows:
- **Concatenate** the key and the Hexadecimal Neutrosophic System string: "password1236A4F5D7E".
- **Apply** a hash function (e.g., SHA-256) to the concatenated string:
"c6d9a3e6ca51f5b06c3e2d0c7e6c10954f3d2d7a6e5d2a4c3e4d8d6a1dbbaf5e".
- **Take** the first 16 characters of the hash output to get the encoded string: "c6d9a3e6ca51f5b0".
4. **Output** the encoded Hexadecimal Neutrosophic System string (e.g., "c6d9a3e6ca51f5b0").

Therefore, this algorithm demonstrates how data can be encoded using the Hexadecimal Neutrosophic System, and how the encoding can be secured using a key to ensure the confidentiality and integrity of the data during transmission.

Example

An example of how the Hexadecimal Neutrosophic System encoding algorithm could be used in practice:

Let's say that Soso wants to send a password, "mysecretpassword", to Toto securely using the Hexadecimal Neutrosophic System encryption algorithm. She follows the encoding algorithm:

1. Convert the password into a vector of degrees of truth, indeterminacy, and falsity: [0.9, 0.1, 0.0].
2. Convert the Neutrosophic vector to a base-16 Hexadecimal Neutrosophic System string: "6A4F5D7E".
3. Encode the Hexadecimal Neutrosophic System string using a key, such as "password123":
- Concatenate the key and the Hexadecimal Neutrosophic System string: "password1236A4F5D7E".
- Apply a hash function (e.g., SHA-256) to the concatenated string:
"c6d9a3e6ca51f5b06c3e2d0c7e6c10954f3d2d7a6e5d2a4c3e4d8d6a1dbbaf5e".
- Take the first 16 characters of the hash output to get the encoded string: "c6d9a3e6ca51f5b0".
4. Output the encoded string: "c6d9a3e6ca51f5b0".

Soso sends the encoded string, "c6d9a3e6ca51f5b0", to Toto over a secure communication channel.

Toto receives the encoded string and wants to decode it to retrieve the original password. He has the key, "password123". Toto follows the decoding algorithm:

1. **Decode** the encoded string using the key:
   - **Concatenate** the key and the encoded string: "password123c6d9a3e6ca51f5b0".

- **Apply** a hash function (e.g., SHA-256) to the concatenated string: "c6d9a3e6ca51f5b06c3e2d0c7e6c10954f3d2d7a6e5d2a4c3e4d8d6a1dbbaf5e".
- **Take** the first 16 characters of the hash output to get the Hexadecimal Neutrosophic System string: "6A4F5D7E".

2. **Convert** the Hexadecimal Neutrosophic System string back into a vector: [0.9, 0.1, 0.0].
3. **Output** the original password: "mysecretpassword".

Therefore, the Hexadecimal Neutrosophic System encoding algorithm allows Soso to securely send the password to Toto, and Toto is able to retrieve the original password using the key. An algorithm to convert given data to neutrosophic form:

This algorithm converts a set of data points to neutrosophic form by assigning each data point to a neutrosophic triplet consisting of a truth value, an indeterminacy value, and a falsity value. The truth value of a data point represents the degree to which it is true, the falsity value represents the degree to which it is false, and the indeterminacy value represents the degree to which it is neither true nor false.

The algorithm first initializes the truth value, indeterminacy value, and falsity value of each data point to 0 and assigns the indeterminacy value to 0.5. Then, it calculates the minimum and maximum values of the data set. For each data point, the algorithm calculates its truth value using the formula (di - min_val) / (max_val - min_val), where di is the data point value, and min_val and max_val are the minimum and maximum values of the data set, respectively. The falsity value is calculated using the formula (max_val - di) / (max_val - min_val). Finally, it assigns the calculated truth value, indeterminacy value (0.5), and falsity value to the corresponding neutrosophic triplet (ti, 0.5, fi). The algorithm returns the set of neutrosophic triplets NT, which represents the neutrosophic form of the input data set.

Algorithm 12. The steps of setting neutrosophic triples

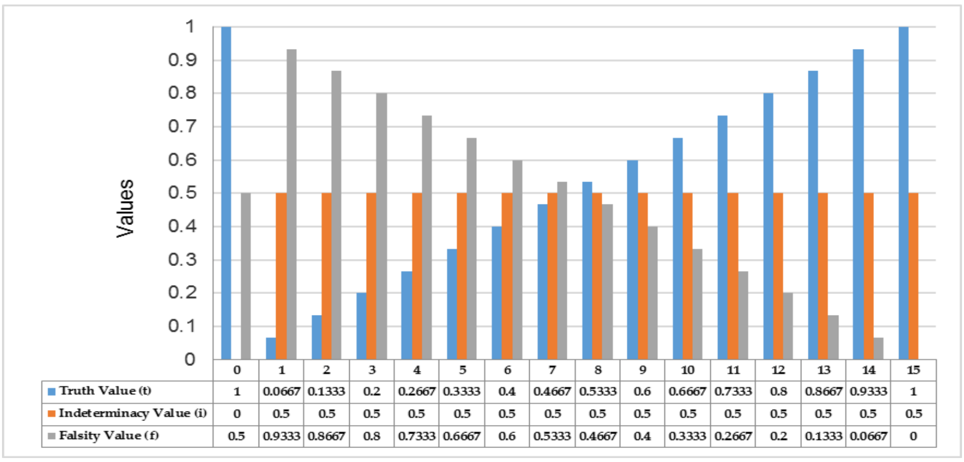
```
Input: a set of data points D = {d1, d2, ..., dn}
Output: a set of neutrosophic triplets NT = {(t1, i1, f1), (t2, i2, f2), ..., (tn, in, fn)}
1. For each data point di in D do the following:
    a. Initialize the truth value, indeterminacy value, and falsity value to 0.
    b. Assign the indeterminacy value to 0.5.
2. Let min_val be the minimum value in D, and max_val be the maximum value in D.
3. For each data point di in D do the following:
    a. Calculate its truth value using the formula: (di - min_val) / (max_val - min_val).
    b. Calculate its falsity value using the formula: (max_val - di) / (max_val - min_val).
    c. Assign the calculated truth value, indeterminacy value (0.5), and falsity value to the corresponding neutrosophic triplet (ti, 0.5, fi).
4. Return the set of neutrosophic triplets NT.
```

Table (9) shows the conversion of the hexadecimal values in D to their corresponding decimal values, along with the calculated truth values, indeterminacy values, and falsity values for each data point in the set:

Figure (6) shows the conversion of hexadecimal and decimal values to their corresponding truth, indeterminacy, and falsity values in the Neutrosophic logic system. The Neutrosophic logic system is a generalization of fuzzy logic that allows for degrees of truth, indeterminacy, and falsity to be represented simultaneously. Each hexadecimal and decimal value is associated with a truth-value (t), an indeterminacy value (i), and a falsity value (f), which together represent the degree of truth, indeterminacy, and falsity of the value in the Neutrosophic logic system. For example, the hexadecimal value "0" and the decimal value "0" have a truth-value of 1, an indeterminacy value of 0, and a falsity value of 0.5, indicating that it is completely true and somewhat false. The table provides a useful reference for converting between different number systems and the Neutrosophic logic system.

Table (9). The Hexadecimal, Decimal, Indeterminacy, and Falsity Value.

Hexadecimal	**Decimal**	**Truth Value (t)**	**Indeterminacy Value (i)**	**Falsity Value (f)**
0	0	1	0	0.5
1	1	0.0667	0.5	0.9333
2	2	0.1333	0.5	0.8667
3	3	0.2	0.5	0.8
4	4	0.2667	0.5	0.7333
5	5	0.3333	0.5	0.6667
6	6	0.4	0.5	0.6
7	7	0.4667	0.5	0.5333
8	8	0.5333	0.5	0.4667
9	9	0.6	0.5	0.4
A	10	0.6667	0.5	0.3333
B	11	0.7333	0.5	0.2667
C	12	0.8	0.5	0.2
D	13	0.8667	0.5	0.1333
E	14	0.9333	0.5	0.0667
F	15	1	0.5	0



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
■ Truth Value (t)	1	0.0667	0.1333	0.2	0.2667	0.3333	0.4	0.4667	0.5333	0.6	0.6667	0.7333	0.8	0.8667	0.9333	1
■ Indeterminacy Value (i)	0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
■ Falsity Value (f)	0.5	0.9333	0.8667	0.8	0.7333	0.6667	0.6	0.5333	0.4667	0.4	0.3333	0.2667	0.2	0.1333	0.0667	0

Figure 6. The Histogram of Truth Values and Indeterminacy and Falsity

18. Conclusions

The use of neutrosophic mathematics offers a new approach to dealing with uncertainty and ambiguity in numerical systems and cryptography. The development and application of neutrosophic number systems and codes have the potential to improve the accuracy and reliability of numerical data analysis and modeling. Further research is needed to explore the practical applications and benefits of neutrosophic mathematics in various domains, including cryptography, machine learning, and finance. The development of software tools for working with neutrosophic number systems could also facilitate the adoption of this approach in various fields. The impact of neutrosophic mathematics on education and curriculum could also be studied, providing students with a deeper understanding of mathematical concepts and their practical applications. Overall, the use of neutrosophic mathematics presents a promising and innovative approach that could lead to new insights and advancements in the field of mathematics and beyond.

Acknowledgement

This research is supported by the Neutrosophic Science International Association (NSIA) in both of its headquarters at New Mexico University and its Iraqi branch at Telafer University, for more details about (NSIA), see the URL http://neutrosophicassociation.org/.

Conflicts of Interest: The authors declare no conflict of interest.

References:

1. Tan, W. K., & Abdullah, S. (2017). Neutrosophic number and its applications. Journal of Intelligent & Fuzzy Systems, 32(1), 89-98.
2. Smarandache, F. (1998). Neutrosophy and neutrosophic logic, set, probability, and statistics. Rehoboth: American Research Press.

3. Smarandache, F., & Dezert, J. (Eds.). (2015). Advances and applications of DSmT for information fusion. Springer.
4. Wang, J. Q., & Smarandache, F. (2017). Introduction to neutrosophic statistics. Neutrosophic Science International Journal, 4(11), 1-42.
5. Smarandache, F. (2020). Neutrosophic set and logic: Theory and applications in computing. Springer.
6. Smarandache, F., & Broumi, S. (2021). Neutrosophic logic and its applications. Studies in Systems, Decision and Control, 290.
7. Edalatpanah, S. A. (2020). Systems of neutrosophic linear equations. Neutrosophic Sets and Systems, 33, 95-102.
8. Hamza, A., et al, E. (2020). Cryptography in terms of triangular neutrosophic numbers with real life applications. International Journal of Neutrosophic Science, 11(1), 39-52.
9. Smarandache, F., & Vasantha Kandasamy, W. B. (2017). The applications of fusion neutrosophic number theory in public key cryptography and the improvement of RSA algorithm. Neutrosophic Sets and Systems, 18(1), 3-9.
10. Salama, A. A., & Smarandache, F. (2014). Neutrosophic crisp set theory. Neutrosophic Sets and Systems, 5, 27-35.
11. Salama, A. A., El-Ghareeb, H. A., Manie, A. M., & Smarandache, F. (2014). Introduction to develop some software programs for dealing with neutrosophic sets. Neutrosophic Sets and Systems, 3, 51-52.
12. Elwahsh, H., Gamal, M., Salama, A., & El-Henawy, I. (2018). A novel approach for classifying MANETs attacks with a neutrosophic intelligent system based on genetic algorithm. Security and Communication Networks, 2018, 1-7.
13. ElWahsh, H., Gamal, M., Salama, A., & El-Henawy, I. (2018). Intrusion detection system and neutrosophic theory for MANETs: A comparative study. Neutrosophic Sets and Systems, 23, 16-22.
14. Ibrahim Yasser, A., Twakol, A., Abd El-Khalek, A. A., Samrah, A., & Salama, A. A. (2020). COVID-X: Novel health-fog framework based on neutrosophic classifier for confrontation Covid-19. Neutrosophic Sets and Systems, 35, 1-21.
15. Belal Amin, A. A. Salama, I. M. El-Henawy, Khaled Mahfouz, & Mona G. Gafar (2021). Intelligent neutrosophic diagnostic system for cardiotocography data. Computational Intelligence and Neuroscience, 2021, 1-12.
16. Alhasan, K. F., Salama, A. A., & Smarandache, F. (2021). Introduction to neutrosophic reliability theory. International Journal of Neutrosophic Science, 15(1), 52-61.
17. Ibrahim Yasser, A., Abd El-Khalek, A. A., Twakol, A., Abo-Eldin, M. E., Salama, A. A., & Khalifa, F. (2021). A hybrid automated intelligent COVID-19 classification system based on neutrosophic logic and machine learning techniques using chest X-ray images. In Advances in Data Science and Intelligent Data Communication Technologies for COVID-19 (Vol. 378). Springer Nature.
18. Dantzig, T. (1954). Number: The language of science. New York: The Free Press.
19. Cajori, F. (1916). A History of Mathematical Notations. Dover Publications.
20. Knuth, D. E. (1992). Two notes on notation. American Mathematical Monthly, 99(5), 403-422.
21. Smarandache, F. (2002). A unifying field in logics: Neutrosophic logic. Neutrosophy, neutrosophic set, neutrosophic probability, and neutrosophic statistics. Multiple-Valued Logic, 8(3), 297-384.
22. Smarandache, F. (2003). Neutrosophic set—a generalization of the intuitionistic fuzzy set. International Journal of Pure and Applied Mathematics, 4(1), 109-129.
23. Smarandache, F. (2005). Neutrosophy, neutrosophic logic, set, and probability. American Research Press.
24. Smarandache, F. (2006). Neutrosophic set—a tool for generalized uncertain reasoning. International Journal of Computers, Communications & Control, 1(2), 1-22.
25. Smarandache, F., & Negoita, C. V. (1998). A unifying field in sets, logic, topology, algebra, and other branches of mathematics. American Research Press.
26. Smarandache, F.(2016). Neutrosophic triplets. Neutrosophic Sets and Systems, 14, 1-4.

27. Smarandache, F. (2021). Neutrosophic soft sets and their applications. Neutrosophic Sets and Systems, 38, 1-6.
28. Huda E. Khalid, "An Original Notion to Find Maximal Solution in the Fuzzy Neutrosophic Relation Equations (FNRE) with Geometric Programming (GP)", Neutrosophic Sets and Systems, vol. 7, 2015, pp. 3-7.
29. Huda E. Khalid, "The Novel Attempt for Finding Minimum Solution in Fuzzy Neutrosophic Relational Geometric Programming (FNRGP) with (max, min) Composition", Neutrosophic Sets and Systems, vol. 11, 2016, pp. 107-111.
30. Huda E. Khalid, F. Smarandache, A. K. Essa, (2018). The Basic Notions for (over, off, under) Neutrosophic Geometric Programming Problems. Neutrosophic Sets and Systems, 22, 50-62.
31. Huda E. Khalid, (2020). Geometric Programming Dealt with a Neutrosophic Relational Equations Under the ($\boldsymbol{max} - \boldsymbol{min}$) Operation. Neutrosophic Sets in Decision Analysis and Operations Research, chapter four. IGI Global Publishing House.
32. Huda E. Khalid, "Neutrosophic Geometric Programming (NGP) with (max-product) Operator, An Innovative Model", Neutrosophic Sets and Systems, vol. 32, 2020.
33. Huda E. Khalid, F. Smarandache, A. K. Essa, (2016). A Neutrosophic Binomial Factorial Theorem with their Refrains. Neutrosophic Sets and Systems, 14, 50-62.

Received: 15 Dec, 2023 Accepted: 20 Mar, 2024