

# Novel Joint Cybersecurity-Channel Coding Scheme via Homeomorphism and Nano Controlled-Switching Lattice Grids, Part I: New Results

Anas N. Al-Rabadi

Department of Computer Engineering, School of Engineering, The University of Jordan, Amman, Jordan  
Email: alrabadi@yahoo.com, a.alrabadi@ju.edu.jo

**How to cite this paper:** Al-Rabadi, A.N. (2026) Novel Joint Cybersecurity-Channel Coding Scheme via Homeomorphism and Nano Controlled-Switching Lattice Grids, Part I: New Results. *Journal of Computer and Communications*, 14, 93-114.  
<https://doi.org/10.4236/jcc.2026.149007>

**Received:** July 28, 2026

**Accepted:** September 27, 2026

**Published:** September 30, 2026

Copyright © 2026 by author(s) and Scientific Research Publishing Inc.  
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).  
<http://creativecommons.org/licenses/by/4.0/>



Open Access

## Abstract

A new homeomorphic joint cybersecurity-channel coding method is introduced for the first time in this article. Channel coding plays a crucial role in contemporary wireless networking and data transmission, as it is essential for mitigating data errors that arise during data transmissions due to the inevitable presence of channel noise, thereby ensuring increased data integrity and thus enhancing reliability. Further, cybersecurity is highly important in modern wireless data transaction since wireless networks are highly exposed to malicious attacks and eavesdropping, and thus ensuring higher levels of confidentiality, integrity and availability. Logic homeomorphism describes properties-preserving mapping which is bijective. In addition to its necessity in advanced computing paradigms such as within quantum computing, it is introduced for the first time in this article that logic homeomorphism can be used to achieve joint cybersecurity-channel coding design. Further, it is also introduced that the property of logic homeomorphism is important within cybersecurity since one can achieve higher confidentiality using new symmetric key  $k_s$  and the corresponding key-generated homeomorphic mapping Algorithm<sub>m</sub>, and within channel coding since it is shown that homeomorphism relationship between data can be used for further correction within many-error situations that are usually uncorrectable. This first part of the article will introduce new results of novel homeomorphic joint cybersecurity-channel coding scheme that will be effectuated in the second part of the article using hierarchical design via regular lattice grids and nano-based carbon multiplexers where the implementation of an important class of homeomorphic joint cybersecurity-Viterbi decoder will be demonstrated. The introduced method of new joint cybersecurity-channel coding scheme will be useful for improving implemented

ASIC system characteristics of confidentiality and data integrity, in addition to the hierarchical implementation of nano-based regular lattice design that will be useful to enhance other important system characteristics such as speed improvement, size reduction and power minimization.

## Keywords

Cybersecurity, Channel Coding, Logic Homeomorphism

---

## 1. Introduction

The vector of input states can be uniquely reconstructed from the vector of output states in a  $(k, k)$  homeomorphic circuit, which has the same number of  $k$  inputs and  $k$  outputs and is a one-to-one mapping between the vectors of inputs and outputs [1]-[6]. Additionally, a  $(k, k)$  conservative circuit includes  $k$  inputs and  $k$  outputs, as well as the same number of input and output values. Because it represents the physical law of energy preservation by maintaining the same number of values in inputs and outputs—no energy can be created or destroyed, but it can be changed from one form to another—the conservative feature is significant [1]-[6]. Therefore, the law of energy preservation is essentially incorporated into the logic architecture of circuits and systems in conservative logic. Consequently, logic conservativeness is both information lossless and energy lossless, while logic homeomorphism is information lossless. The speed-to-power ratio doesn't improve much in non-homeomorphic systems after a certain threshold because, typically, higher speed (better performance) results in higher power consumption. However, this isn't always the case in the nanoscale domain; for instance, processing speed in nanoscale quantum computing systems is very high because of the properties of superposition and entanglement, and power consumption is inversely very low [1]-[6]. As a result, and since several effective existing and upcoming components of nanoscale computers need to be homeomorphic, particularly in quantum computing, homeomorphic computing is becoming increasingly prevalent in the design of traditional, compact and universal circuits and systems [1] [6].

Since closed nanoscale quantum systems obey the unitary evolution and therefore they are intrinsically homeomorphic, motivations to implement circuits and systems using logic homeomorphism within nanoscale systems, such as in quantum computing, would include the important advantages of higher speed, lower power and smaller size [1] [6] as follows:

- 1) speed (performance): if the characteristics of superposition and entanglement in nanoscale quantum mechanical systems can be effectively utilized in the development of computational circuits and systems, substantial improvements in computational speed will be realized.

- 2) power: theoretical internal computations in homeomorphic systems con-

sume no power, as demonstrated [5] that information is physical; the heat dissipated from each non-homeomorphic bit operation is expressed as  $K \cdot T \ln(2)$ , with  $K$  being the Boltzmann constant and  $T$  representing the operating temperature, emphasizing that a necessary yet insufficient condition for power conservation in any physical system is that all circuits must be logically homeomorphic. Consequently, various technologies have been researched to achieve logic homeomorphism in hardware, including adiabatic CMOS VLSI circuit design and nano circuits and systems. Completely homeomorphic digital systems will, in theory, eradicate power consumption through three criteria: a) logical homeomorphism: the input states vector can be uniquely reconstructed from the output states vector at all times, b) physical homeomorphism: the physical switch functions in reverse as well as forward, and c) “ideal-like” switches devoid of parasitic resistances.

3) size: current trends in nanoscale hardware developments are moving towards atomic dimensions, where quantum characteristics like entanglement and superposition become crucial.

Cybersecurity involves safeguarding digital systems, networks, devices, software, databases and information from cyberattacks, theft or harm [7]–[9]. Overall, data must be protected from threats, particularly in sectors like banking, manufacturing, and defense. To ensure security, information—like files on computers—must be concealed from unauthorized access (confidentiality), safeguarded against unauthorized alterations (integrity), and accessible to an authorized user when required (availability), referred to in cybersecurity as the C.I.A. triad. The execution of these three requirements is distinct and more difficult. Confidentiality is likely the most prevalent element of information security, relating to how information is stored and transmitted. Integrity signifies that modifications to information should only be executed by permitted individuals and through sanctioned methods, where a breach of integrity isn’t always due to malicious intent; disruptions in the system, like a power surge, can similarly lead to unintended alterations in certain information. For instance, details in a bank require frequent updates; when a customer adds or takes out money, the customer’s account balance must be properly adjusted. Availability is crucial because the data generated and stored by an organization must be accessible to authorized individuals; if information is not available, it is of no value. Typically, information must be regularly updated, indicating it should be accessible to authorized parties; the inaccessibility of information can be as detrimental to an organization as a deficiency in confidentiality or integrity. For instance, one might envision the consequences for a bank if users were unable to reach their accounts for transactions. Overall, all the aims of the C.I.A. security triad can be compromised by breaches and attacks. Therefore, the real execution of security objectives requires security methods that can be broad like cryptography and others that are specific like steganography [7]–[9]. Cryptography generally involves the process of converting messages to ensure they are secure and resistant to external threats, typically comprising three separate mechanisms: 1) symmetric-key encryption, which employs a single secret key

for both encryption and decryption, as in DES and AES algorithms, 2) asymmetric-key encryption, which utilizes two keys—a public key for encryption and a private key for decryption, exemplified by RSA and Elliptic Curve algorithms, and 3) hashing, where a fixed-length message digest is generated from a variable-length message, with both the original message and the digest transmitted to the recipient in applications that specifically utilize checksums such as in data integrity tasks.

As previously mentioned, security goals in modern cybersecurity comprise of confidentiality, integrity, and availability. The classification of security attacks with relation to security goals can be categorized as follows [8] [9]: 1) the set of passive attacks {snooping, traffic analysis} are threats to confidentiality, 2) the set of active attacks {modification, masquerading, replaying, repudiation} are threats to integrity, and 3) the set of active attacks {denial of service} is a threat to availability. On the other hand, the relationship between security services and security mechanisms can be categorized as follows [8] [9]: 1) the set of security mechanisms {encipherment, routing control} can be used to provide the security service of data confidentiality, 2) the set of security mechanisms {encipherment, digital signature, data integrity} can be used to provide the security service of data integrity for anti-change and anti-replay, 3) the set of security mechanisms {encipherment, digital signature, authentication exchanges} can be used to provide the security service of authentication for peer entity and data origin, 4) the set of security mechanisms {digital signature, data integrity, notarization} can be used to provide the security service of non-repudiation for proof of origin and proof of delivery, and 5) the set of security mechanisms {access control mechanism} can be used to provide the security service of access control. While security mechanisms are only a theoretical guide to implement security, security techniques are the actual implementation of security goals that include the general technique of cryptography which constitutes of encipherment (both symmetric-key and asymmetric-key) and hashing, and the specific technique of steganography.

One of the biggest risks to computer security is unauthorized access to a computer system or network [7]-[9]. In order to avoid or minimize harm, intrusion detection systems have been created to provide early warning of an intrusion. Finding anomalous activity patterns or activity patterns that are known to be associated with intrusions is the goal of intrusion detection. Password management is an important example of intrusion prevention which aims to keep unauthorized individuals from accessing other people's passwords. For example, malicious software can be inserted into a system for spying or to cause a harm. A virus, for example, is a program that infects other software by modifying it to include a copy of itself, allowing it to spread further. Another example is a worm, which is a self-replicating program that sends copies of itself to other computers across network connections; once it reaches a new system, the worm can activate to replicate and propagate again, while typically performing additional unwanted functions. As an application example of security programs, PGP is a free and open-source email

security program that uses digital signatures for authentication, symmetric block encryption for confidentiality, ZIP algorithm for compression, radix-64 encoding scheme for email compatibility, and segmentation and reassembly to handle lengthy emails [7]-[9], where PGP includes tools for managing public-key certificates and creating a public-key trust model.

Typically, during data exchanges between two connected nodes, noise interferes and distorts the transmitted data messages, resulting in the reception of noisy corrupted messages [2] [3] [10]-[12]. Typically, the communication channel is where the corrupting noise originates. Consequently, classical error correction of transmitted data [10]-[12] and homeomorphic error correction of transmitted batches of data [2] [3] are crucial activities in scenarios where channel noise is present. Several methods have been traditionally employed to address classical error detection and correction (EDAC) issues: one approach involves utilizing linear error-correcting codes like Hamming, Reed-Solomon, Low-Density Parity-Check (LDPC), Polar and Golay codes, while another approach leverages different coding strategies that perform optimally for specific types of statistical noise distributions as seen in the Viterbi and Turbo algorithms.

The primary contributions of this first part of the article include the novel demonstration that logic homeomorphism can facilitate joint cybersecurity-channel coding design, along with new examples that highlight the significance of logic homeomorphism in cybersecurity, allowing for greater confidentiality through new symmetric key and the related key-generated homeomorphic mapping, as well as in channel coding where the homeomorphic relationship among data is utilized for enhanced correction in many-error scenarios that typically cannot be corrected. The newly obtained results in the proposed homeomorphic joint cybersecurity-channel coding scheme, presented here for the first time in this first part of the article, will be utilized in the second part of the article for the relevant architectural hierarchical design employing regular lattice grids through nano-based carbon controlled-switching methods. **Figure 1** illustrates the specific layers of the proposed hierarchical implementation discussed in this two-part article.

| <b>Homeomorphic Joint Cybersecurity – Channel Coding Computation</b> |
|----------------------------------------------------------------------|
| <b>Lattice Grid Functional Synthesis</b>                             |
| <b>Circuits and System Design</b>                                    |
| <b>Carbon Controlled-Switching Devices</b>                           |
| <b>Solid-State, EM and Field-Emission Physics</b>                    |
| <b>Galois Field Algebra</b>                                          |

**Figure 1.** Hierarchy of the introduced system implementation.

The first part of this article is structured as follows: Fundamentals of logic homeomorphism are discussed in Section 2. Section 3 provides an overview of channel coding that utilizes forward error correction (FEC). Homeomorphic channel coding utilizing homeomorphic convolutional encoding (on the sender's side) and the related homeomorphic Viterbi decoding (on the receiver's side) is dis-

cussed in Section 4. The innovative approach to joint cybersecurity-channel coding technique employing logic homeomorphism, which uses a novel symmetric key  $k_s$  along with the associated key-generated homeomorphic mapping Algorithm<sub>m</sub>, is introduced and fully illustrated in Section 5. Section 6 contains the conclusions.

## 2. Logic Homeomorphism

A logically homeomorphic circuit, denoted as  $(n, k)$ , typically features  $n$  inputs and  $k$  outputs, establishing a one-to-one correspondence between the input and output vectors. Consequently, the input states vector can be uniquely derived from the output states vector [1]-[6]. Hence, a homeomorphic map is a bijective function that is both 1) injective (one-to-one) and 2) surjective (onto). The ancillary outputs are required solely for achieving logic homeomorphism; these outputs serve as the basis for constructing a homeomorphic map (see Example 1), resulting in logically homeomorphic systems that do not lose information. For instance, closed nanoscale quantum systems follow unitary evolution, making them intrinsically homeomorphic. From a geometric perspective, attaining logical homeomorphism results in value-driven space division, which creates spatial sections of distinct values. Algebraically, logic homeomorphism results in bijective mapping with multiple inputs and outputs (MIMO). An algorithm known as homeomorphic Boolean function (HBF) generates a MIMO homeomorphic function from its non-homeomorphic version in the following manner [1].

---

### Algorithm HBF

---

- 1) To get I/O logic homeomorphism, add adequate number of ancillary outputs such that number of outputs matches the number of inputs. Allocate a new column for each ancillary output.
  - 2) To get the first ancillary output, allocate a constant  $K_1$  to half of the cells in the corresponding table column (e.g., ones) and the second half as another constant  $K_2$  (e.g., zeros).
  - 3) **For** the next ancillary output, **If** non-homeomorphism still exists, **Then** allocate for same output tuples values which are half zeros and half ones, **Else** allocate a constant for the rest.
  - 4) **Do** step 3 till all map items are logically homeomorphic.
- 

**Example 1.** The subsequent I/O map is a logically non-homeomorphic map:

| $c$ | $d$ | $W$ |
|-----|-----|-----|
| 0   | 0   | 0   |
| 0   | 1   | 1   |
| 1   | 0   | 1   |
| 1   | 1   | 0   |

By implementing the HBF algorithm, **Figure 2** illustrates two possible (2, 2) homeomorphic maps.

| $c$ | $d$ | $W$ | $W_1$ |
|-----|-----|-----|-------|
| 0   | 0   | 0   | 1     |
| 0   | 1   | 1   | 0     |
| 1   | 0   | 1   | 0     |
| 1   | 1   | 0   | 1     |

(a)

| $c$ | $d$ | $W$ | $W_1$ |
|-----|-----|-----|-------|
| 0   | 0   | 0   | 0     |
| 0   | 1   | 1   | 1     |
| 1   | 0   | 1   | 0     |
| 1   | 1   | 0   | 1     |

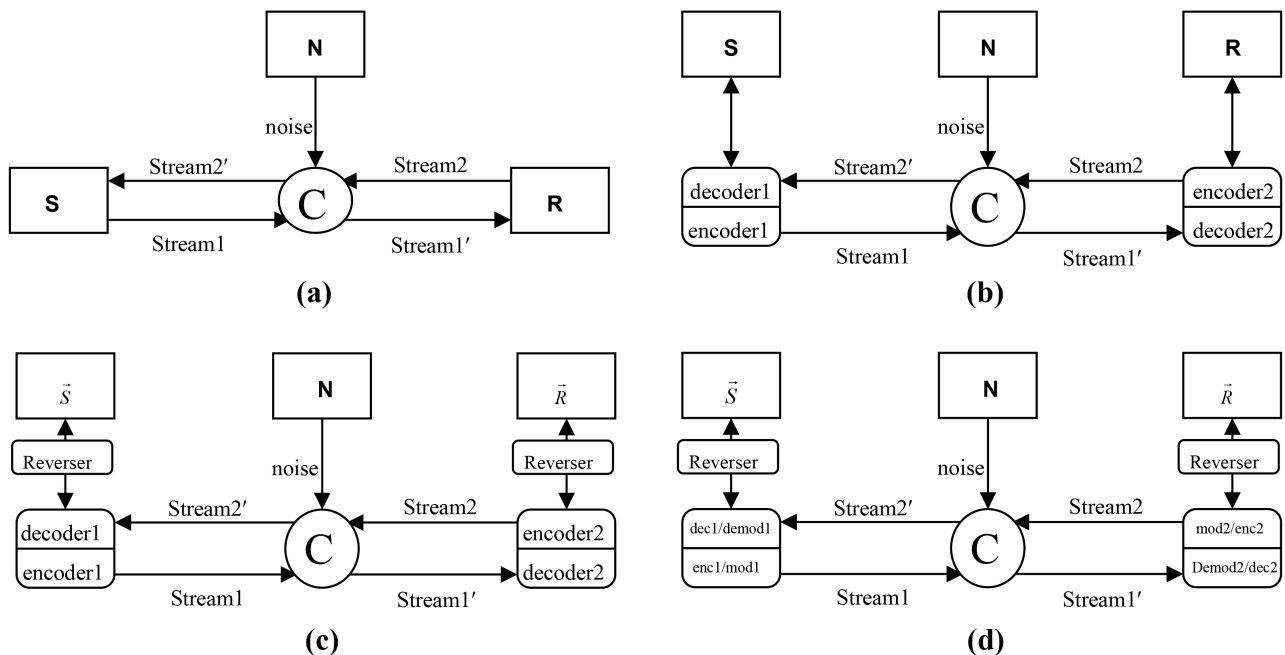
(b)

**Figure 2.** Two possible homeomorphic maps for the logically non-homeomorphic map in Example 1.

### 3. Channel Coding: Forward Error Correction

In data communication, noise is typically produced by the medium through which transmitted data is exchanged. This type of noise distorts sent messages from one side, leading to the reception of corrupted messages on the opposite end. To address the issue of retrieving an accurate message from its corrupted version, it is essential to model the noise and implement suitable channel coding methods accordingly [2] [3] [10]-[12]. Several forward error correction (FEC) channel coding techniques have been suggested, with a significant category being the convolutional channel coding approaches like Turbo coding and Viterbi coding methods.

**Figure 3** depicts the modeling of data communication amid noise, the resolution of the noise issue through an encoder/decoder system, and the application of a block known as the reverser to attain logic homeomorphism during data transactions [2].



**Figure 3.** Modeling data transactions in the presence of noise: (a) a model depicting noisy data communication with C as the channel, N representing noise, S as the sender, and R as the receiver, (b) a model addressing the noise issue through FEC encoder/decoder schemes, (c) the implementation of logic homeomorphism utilizing a reverser block for data transactions, and (d) homeomorphic reverser-based communication model with joint coding and modulation design such as in trellis-coded modulation (TCM).

Both sides of the two nodes in the system depicted in **Figure 3** comprise three key components: 1) encoding (for instance, creating a convolutional code through a convolutional encoder) to produce the encoded transmitted data, 2) channel noise  $N$ , and 3) decoding (for example, obtaining the accurate output via the appropriate convolutional decoder) to yield the correct decoded data at the receiver. Typically, the block coding encoder takes a  $k$ -bit message block and produces an  $n$ -bit code word, meaning code words are created one block at a time, and the entire message block has to be stored before the corresponding code word is produced. Conversely, message bits are received one at a time instead of using blocks, making it inconvenient to utilize a buffer, where convolutional coding is employed in this method, wherein a convolutional coder produces redundant bits through modulo-2 convolutions. The binary convolutional encoder functions as a finite state machine (FSM) that includes an  $M$ -stage shift register linked to  $n$  modulo-2 adders and a multiplexer to serialize the adders' outputs, whereby an  $L$ -bit message sequence produces a coded output sequence with a length of  $n(L + M)$  bits [10]-[12].

**Definition 1.** For an  $L$ -bit message sequence,  $M$ -stage shift register,  $n$  modulo-2 adders, and a generated coded output sequence of length  $n(L + M)$  bits, the code rate  $r$  is calculated as:

$$r = \frac{L}{n(L + M)} \text{ bit/symbol}$$

and for the typical case of  $L \gg M$ , the code rate reduces to  $r \approx \frac{1}{n}$  bit/symbol.

**Definition 2.** The constraint length  $K$  of a convolutional code represents the number of shifts during which a single message bit can affect the encoder output. Therefore, for an encoder featuring an  $M$ -stage shift register, the total number of shifts needed for a message bit to enter the shift register and subsequently exit is  $K = M + 1$ .

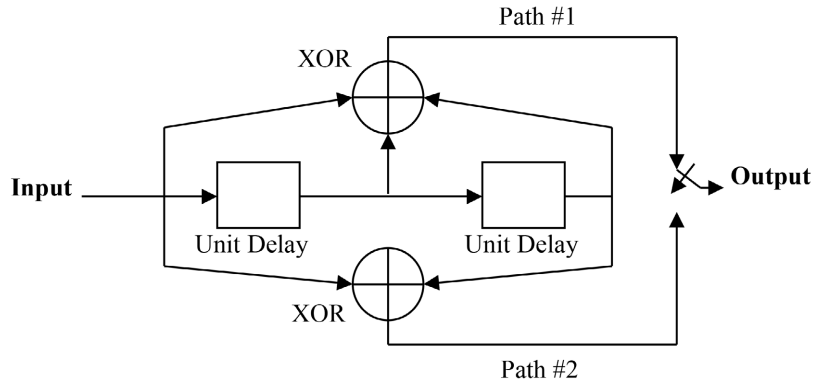
A binary convolutional code can be generated in general with code rate  $r = \frac{k}{n}$ , that shows the efficiency of the code, where  $k$  input bits are mapped into  $n$  output bits during each encoder cycle, or can be generated in special case with code rate  $r = \frac{1}{n}$  where the encoder takes one input bit and generates  $n$  output bits. An example of a convolutional encoder with constraint length  $K = 3$  and rate  $= \frac{1}{2}$  is the one shown in **Figure 4**.

The convolutional codes produced by encoders like the one shown in **Figure 4** fall under the category commonly referred to as nonsystematic codes. Every route linking the output to the input of a convolutional encoder can be described through the impulse response, which is defined as the reaction of that route to a "1" applied to its input, with all flip-flops of the encoder initially set to "0." Similarly, we can describe each path by using a generator polynomial defined as the unit-delay transformation of the impulse response. The generator polynomial is

more precisely defined as:

$$r(E) = \sum_{j=0}^N r_j E^j \quad (1)$$

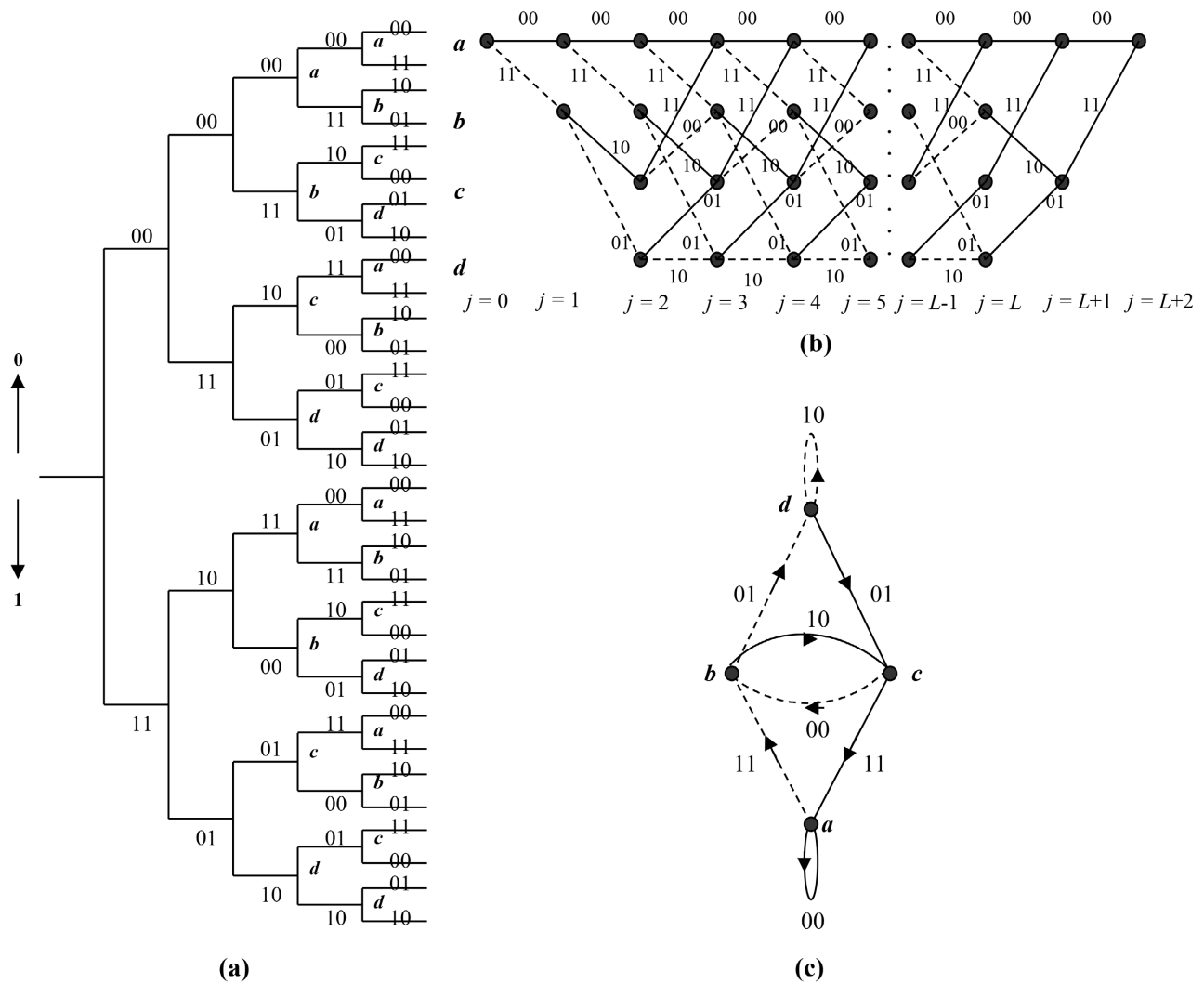
where generator coefficients  $r_i \in \{0,1\}$ , generator sequence  $\{r_0, r_1, \dots, r_N\}$  which is composed of generator coefficients is the impulse response of the corresponding path in the convolutional encoder, and  $E$  is the unit-delay variable.



**Figure 4.** Convolutional encoder with constraint length  $K=3$  and rate  $= 1/2$ , where the flip-flop is a unit-delay element and modulo-2 adder is the logic XOR operation. The rate  $r = 1/2$  shows that the encoder takes in 1 bit and gives out 2 coded bits, constraint length  $K=3$  shows that it uses the current input bit and 2 past input bits, number of memory elements of  $K-1=2$  shows that it needs 2 delay units (flip-flops/registers) to store the past 2 input bits, number of outputs  $n=2$  shows that two modulo-2 adders (XOR gates) combine the current input and delayed bits to create the 2 output bits per clock cycle, and number of states of  $2^{K-1}=4$  shows that the shift registers can hold 4 unique combinations (00, 01, 10, 11).

Typically, a data message sequence that is  $L$  bits long produces an encoded sequence measuring  $n(L+K-1)$  bits. Typically, a terminating sequence consisting of  $(K-1)$  zeros, known as the message tail, is added to the final input bit of the message sequence to reset the shift register to its initial zero state. The convolutional encoder's structural properties (e.g., **Figure 4**) can be visually illustrated in multiple equivalent forms (see **Figure 5**) through: 1) code tree, 2) trellis, and 3) state diagram. The trellis comprises  $(L+K)$  stages, with  $L$  representing the length of the incoming message sequence and  $K$  indicating the constraint length of the code. Consequently, the trellis structure is favored over the code tree structure since the number of nodes at each level of the trellis does not exponentially increase with the number of incoming message bits, but stays constant at  $2^{K-1}$ .

**Figure 5** illustrates the different graphical representations for the convolutional encoder depicted in **Figure 4**, demonstrating that any encoded output sequence can be produced from the associated input message sequence through the following equivalent approaches: 1) convolutional encoder circuit (**Figure 4**), 2) polynomial generator, 3) code tree (**Figure 5(a)**), 4) trellis diagram (**Figure 5(b)**), and 5) state diagram (**Figure 5(c)**).



**Figure 5.** Different representations of the convolutional encoder circuit shown in **Figure 4**: (a) code tree, (b) trellis, and (c) state diagram. The solid line represents an input value of “0,” while the dashed line signifies an input value of “1.” The binary tag on every branch represents the output of the encoder transitioning from one state to another. The encoding of the states can be expressed as  $\{a = 00, b = 10, c = 01, d = 11\}$ .

A significant type of convolutional decoder that utilizes the trellis structure to fix received faulty messages is the Viterbi decoding algorithm [10]–[12]. The Viterbi decoder is a dynamic programming technique employed to determine the maximum-likelihood sequence of hidden states, leading to a series of observed events, especially within hidden Markov models. The Viterbi decoder is an important element within information theory and has been widely utilized across various applications such as speech recognition, keyword detection, computational linguistics, bioinformatics, and in contemporary communications encompassing cellular networks, satellite systems, deep-space communications and wireless local area networks.

The Viterbi decoder is a decoder based on maximum likelihood that is ideal for Additive White Gaussian Noise. This FEC decoding algorithm functions by cal-

culating a metric for each potential path in the trellis diagram. The metric for a given path is determined by calculating the Hamming distance between the coded sequence associated with that path and the received sequence. For two code vectors that contain the same number of elements, the Hamming distance for this pair is defined as the count of positions where their corresponding elements are different. In the context of the Viterbi decoding algorithm, Hamming distance is determined by counting the number of differing bits between the received channel symbol pair and the potential channel symbol pairs with outcomes belonging to the set of values  $\{0, 1, 2\}$ .

Consequently, for every node in the trellis, the Viterbi decoder assesses the two paths leading into the node; the path with the lesser metric is preserved while the other path is eliminated. This calculation is performed for each level  $j$  of the associated trellis within the interval between  $M = K - 1$  that represents the encoder's memory and  $L$  that indicates the length of the incoming message sequence. The routes that are kept are termed survivor or active paths. When the routes leading to a node are assessed and their metrics are determined to be the same, a decision is made by taking a chance (*i.e.*, tossing a fair coin). As mentioned earlier, the Viterbi decoder serves as a maximum likelihood sequence estimator, with the subsequent procedure demonstrating the specific steps for implementing this important decoding algorithm.

---

#### Algorithm Viterbi

---

- 1) *Initialization*: Mark the left-most state of the trellis at level zero as 0.
- 2) *Computation*: Let  $j = 0, 1, 2, \dots$ , and presume at level  $j$  the next is done:
  - a) All survivor paths are known.
  - b) The survivor paths and its metric for each state of the trellis are saved.

**Then**, at level  $(j + 1)$  and **For** all the paths going into each state of the trellis, calculate the metric by summing the metric of the input branches to the metric of the connecting survivor path from level  $j$ . Hence, for each state, determine the path with the minimum metric as the survivor of step  $(j + 1)$ , thus upgrading the calculation.

3) *Finalization*: **Continue** the calculation until the algorithm finishes the forward search through the trellis and thus arrives at the concluding node (*i.e.*, all zero state), at which it decides maximum-likelihood path. **Then**, the sequence of symbols related with that path is discharged to destination as decoded data of received message.

---

**Example 2.** In the convolutional encoder depicted in **Figure 4**, the impulse response for path #1 is (1, 1, 1), while for path #2 it is (1, 0, 1). Therefore, the associated generating polynomials are, respectively:

$$r_1(E) = 1 \cdot E^0 + 1 \cdot E^1 + 1 \cdot E^2 = 1 + E + E^2$$

$$r_2(E) = 1 \cdot E^0 + 0 \cdot E^1 + 1 \cdot E^2 = 1 + E^2$$

For a sequence of messages (101), the  $E$ -domain polynomial form is as follows:

$$d(E) = 1 \cdot E^0 + 0 \cdot E^1 + 1 \cdot E^2 = 1 + E^2$$

Since convolution in the time domain converts to multiplication in the  $E$ -domain, the output polynomial for path #1 and the output polynomial for path #2 are respectively expressed below where addition takes place using modulo-2 arithmetic:

$$e_1(E) = r_1(E)d(E) = (1 + E + E^2) \cdot (1 + E^2) = 1 + E + E^3 + E^4$$

$$e_2(E) = r_2(E)d(E) = (1 + E^2) \cdot (1 + E^2) = 1 + E^4$$

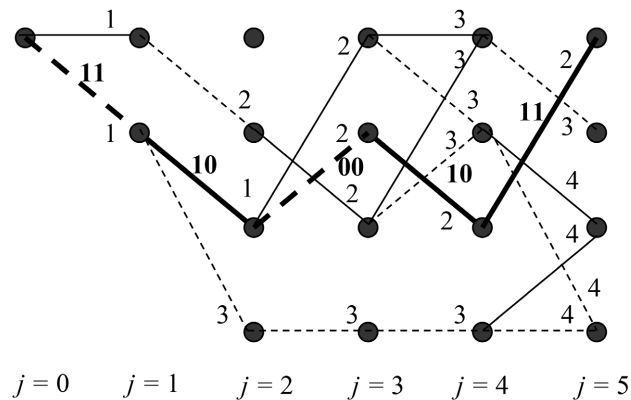
Consequently, the resulting output sequences of paths #1 and #2 are {(11011), (10001)}, respectively. The encoded sequence produced by the convolutional encoder in **Figure 4** is generated by interleaving the two output sequences from paths #1 and #2 in the following manner:

$$e = (11, 10, 00, 10, 11)$$

Let's consider that a noise distorts this sequence, resulting in the following received sequence with interference:

$$e' = (01, 10, 10, 10, 11)$$

Using the Viterbi decoding algorithm, the following is the resulting survivor path as shown in **Figure 6** which generates the correct sent message  $e = (11, 10, 00, 10, 11)$ .



**Figure 6.** The survivors obtained from applying the Viterbi decoding algorithm in Example 2, with the bold path representing the survivor path.

A challenge with using the Viterbi decoding algorithm arises when the received sequence is excessively lengthy. In this scenario, the Viterbi decoding algorithm is utilized on a truncated path memory with a decoding window of at least five times the convolutional code constraint length  $K$ , where it functions on a frame-by-frame basis of the received sequence. The decoding choices made in this manner do not represent a true maximum likelihood, but they can be nearly as effective if the decoding window is sufficiently extensive. An additional challenge is the error-correction ability, as the Viterbi decoder often strives to generate a correctable decoded message from the incoming erroneous message in many-error

situations, where error-correction efficiency relies not only on the error count but also on other elements like error pattern, free distance, and termination assumptions.

#### 4. Forward Error Correction via Homeomorphic Viterbi Decoding

In Section 3, the error correction for transmitted data was addressed for single-input single-output (SISO) systems, whereas this Section introduces homeomorphic error correction for transmitted parallel-based data batches in MIMO systems. This MIMO set of parallel data transactions can be generated by 1) converting input serial data streams into their respective parallel form through a sliding window of constant or variable data length or 2) data that is initially generated in parallel. Homeomorphic logic within a set of data batch processing is noted as the objective is:

$$\mathbf{O}_1 = \mathbf{I}_2 \quad (2)$$

where  $\mathbf{O}_1$  is the unique transmitted output data and  $\mathbf{I}_2$  is the unique received input data.

In MIMO systems, the presence of noise can create errors that might result in non-homeomorphism during data communication (*i.e.*, data mapping that is logically non-homeomorphic). Homeomorphic error-control coding can be executed in software through the HV homeomorphic decoding algorithm, in hardware via nano-based ASIC design, or a combination of both software and hardware methods. The subsequent algorithm, referred to as the Homeomorphic Viterbi (HV) Algorithm, presents the implementation of homeomorphic error correction in sets of transmitted data [2].

---

##### Algorithm HV

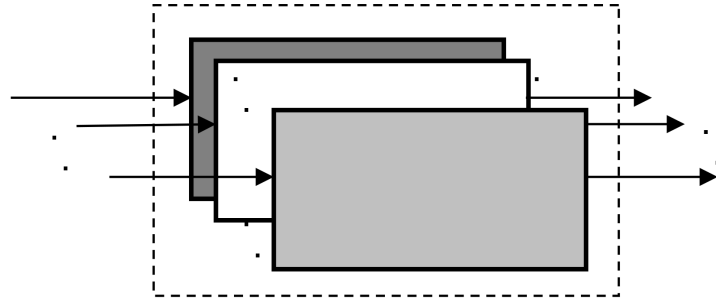
---

- 1) Employ the HBF Algorithm to bijectively encode the transacted batch of parallel data.
  - 2) For a certain convolutional encoder, calculate generator polynomials.
  - 3) For each transacted message within the batch, calculate the encoded data sequence.
  - 4) For each received data, use Viterbi decoder to decode the received (erroneous) data.
  - 5) Generate the total maximum-likelihood trellis from the iterative application of step 4.
  - 6) Generate the corrected transacted batch of data.
- 

Convolutional encoding using the HV Algorithm can be applied 1) sequentially with one convolutional encoder (e.g., from **Figure 4**) or 2) simultaneously (in parallel) using the standard parallel convolutional encoder circuit depicted in **Figure 7**, where multiple convolutional encoders function concurrently to encode messages that are submitted at the same time.

Parallelism in multi-stream data transactions can enable the formation of additional relationships among the transacted data streams, which can then be employed for identifying and correcting errors; logic homeomorphism characteristic through the HV Algorithm generates a homeomorphic relationship among data

batches, and this established homeomorphic mapping can be leveraged for correcting typically uncorrectable errors particularly in complex many-error situations [2].



**Figure 7.** MIMO encoder circuitry for the simultaneous creation of convolutional codes, where each box symbolizes a SISO convolutional encoder like the one depicted in **Figure 4**.

**Example 3.** The following HBF.V1 is a version of the implementation of HBF Algorithm that produces the corresponding logic homeomorphism as follows:

---

**Algorithm HBF.V1**

---

- 1) To get I/O logic homeomorphism, add adequate number of ancillary outputs such that the number of outputs matches the number of inputs. Allocate a new column for each ancillary output.
  - 2) To get the first ancillary output, allocate constant  $K_1$  to upper (half) of the cells in the corresponding table column and the lower (half) as another constant  $K_2$ . For example, allocate constant  $K_1 = 0$  to upper (half) of the column and  $K_2 = 1$  to the lower (half) of the column.
  - 3) For the next ancillary output, **If** non-homeomorphism still occurs, **Then** allocate for same output tuples values which are upper (half) part ones and lower (half) part zeros, and then allocate a constant for the rest which is the one's complement of the previously allocated constant to that remainder.
  - 4) **Do** step 3 till all map items are logically homeomorphic.
- 

For the parallel sent data stream  $\{1, 1, 1\}$ , the implementation of logic homeomorphism—such as through the HBF.V1 Algorithm—yields the corresponding homeomorphic set of data sequences  $\{m_1 = (101), m_2 = (001), m_3 = (011)\}$ . Assume that  $m_1$  and  $m_2$  are accurately decoded while  $m_3$  contains errors because of channel noise. **Figure 8** displays potential tables containing incorrect  $m_3$ .

Note that the incorrect  $m_3$  in **Figures 8(b)-(e)** and **Figure 8(g)**, **Figure 8(h)** can be corrected due to the presence of fewer than triple-errors, while the triple-error case (e.g., **Figure 8(f)**) may be uncorrectable in certain circumstances, influenced by critical factors like the error pattern and free distance. However, the presence of the logic homeomorphism property, exemplified by the HBF Algorithm, contributes information that can be utilized to correct for  $m_3$  as follows: By executing the HBF.V1 Algorithm from right-to-left in **Figure 8(f)**, it is observed that in the second

column (from the right), two “0” cells have been inserted at the top in the accurately received  $m_1$  and  $m_2$  messages, indicating that in the far-right column, the final cell must record as “1” because otherwise, the top two cells in the rightly received  $m_1$  and  $m_2$  messages would need to be “0” and “1,” respectively, to accomplish value-based space-partitioning. Given that the 3<sup>rd</sup> cell of the rightmost column is required to be “1”, it follows that the last cell of the 2<sup>nd</sup> column from the right must also be “1” due to the uniqueness condition as stipulated by the HBF.V1 Algorithm for value-based space-partitioning between the initial two messages  $\{m_1, m_2\}$  and the 3<sup>rd</sup> message  $m_3$ . Subsequently, employing the HBF.V1 Algorithm, the 3<sup>rd</sup> cell of the final column on the right should contain the value “0,” representing the one’s complement (NOT) of the earlier assigned constant “1” in the 3<sup>rd</sup> cell of the 2<sup>nd</sup> column from the right. Therefore, the accurate message  $m_3 = (011)$  is achieved.

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 0 | 1 | 1 |

(a)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 0 | 0 | 1 |

(b)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 1 | 1 | 1 |

(c)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 1 | 0 | 1 |

(d)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 0 | 1 | 0 |

(e)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 1 | 0 | 0 |

(f)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 1 | 1 | 0 |

(g)

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 0 | 0 | 0 |

(h)

**Figure 8.** Tables for possible errors in data stream  $m_3$ : (a) original uncorrupted transmitted  $m_3$ , and (b)-(h) possibilities of corrupted received data stream  $m_3$ .

Example 3 illustrates a straightforward case where the logic homeomorphism characteristic of the HV Algorithm generates an extra homeomorphism relation among the multiple-streams of communicated parallel batches of data, and this established homeomorphic mapping can aid in rectifying uncorrectable errors in various many-error scenarios. These many-error situations can arise from different causes, including fading and burst errors. Typically, fading happens in wireless communication when the strength of the received signal changes and varies because of different factors, like multi-path propagation, resulting in an unreliable data transfer. Moreover, a burst error refers to a series of adjacent corrupted bits in a data stream, and unlike random errors that happen independently, burst errors are interrelated as an error in one bit raises the likelihood of errors in other nearby bits. When a signal undergoes severe fading, its power decreases considerably, potentially corrupting not just a single bit but a group of consecutive transmitted bits, resulting in a burst error situation. Thus, besides employing traditional techniques like interleaving and burst error-correcting codes, and as shown that logic homeomorphism within Viterbi decoding aids in managing complex many-error situations, the proposed error-control design approach utilizing logic homeomorphism can be advantageous for additional correction of error sequences.

## 5. New Joint Cybersecurity-Channel Coding via Logic Homeomorphism

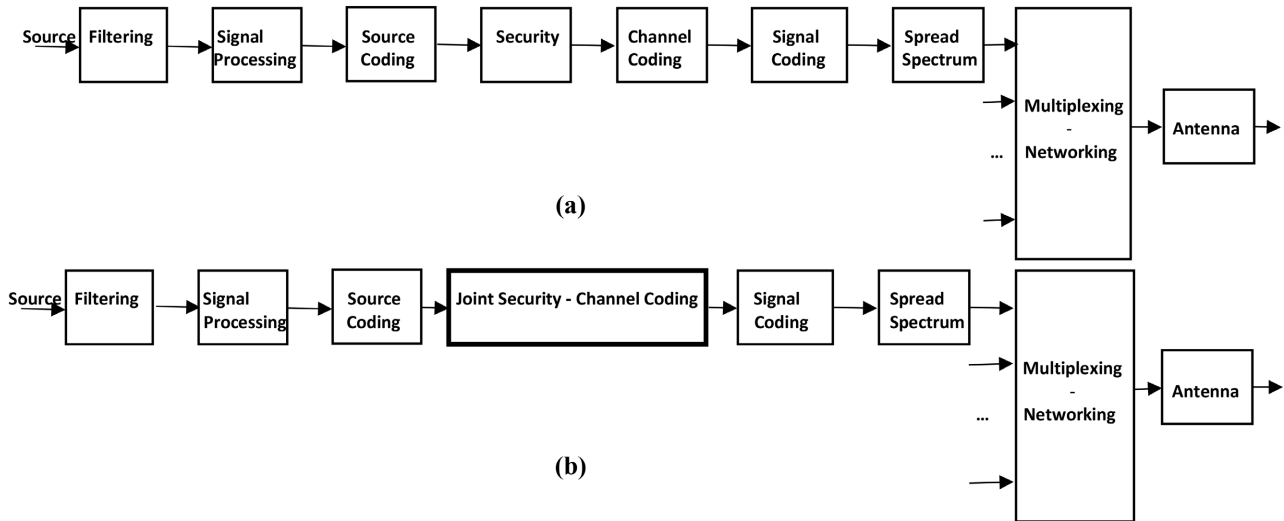
The property of logic homeomorphism will be utilized in this Section for the new joint cybersecurity-channel coding scheme which is introduced for the first time in this article. In general, conventional wireless data transactions usually follow the stages shown in **Figure 9(a)** for data transmission within which: 1) data is originated from a data source such as audio, image, digital systems or sensors, 2) filtering produces cleaned data by removing noise to improve signal-to-noise ratio (S/N) using appropriate methods such as adaptive filtering, 3) signal processing provides signal enhancements such as improving image quality, video quality and information extraction, 4) source coding uses information theory to produce data compression to save bandwidth and achieve higher data rates, 5) data security produces data encryption using cybersecurity techniques, 6) channel coding uses coding theory for error detection and correction (EDAC) to present redundancy into data in order to improve data integrity and thus data reliability, 7) signal coding converts data into the required form which is needed for signal transmission and medium requirements, 8) spread spectrum (SS) produces bandwidth increase of signal to be transmitted to enhance security, interference resistance and anti-jamming, enhancing capacity using CDMA, reducing EMI interference and reducing multipath fading, 9) multiplexing produces data streams from several sources that need to be transmitted over single medium (link) and networking provides device networking over a network architecture such as within IoT or cellular networks, and 10) antenna provides interfacing into the utilized (wireless) transmission medium. For data reception, wireless communication systems usually follow the reverse-order and inverse-function for the corresponding stages shown in **Figure 9(a)**.

While **Figure 9(a)** shows the conventional stages for data transmission, **Figure 9(b)** illustrates the stages of data transmission using the new scheme that joins the two stages of cybersecurity and channel coding to produce a single stage of joint cybersecurity-channel coding via the implementation of logic homeomorphism. The objective for the new joint cybersecurity-channel coding design is to further enhance confidentiality (as will be fully detailed in this Section) and enhance data integrity and therefore reliability (as was illustrated in Section 4).

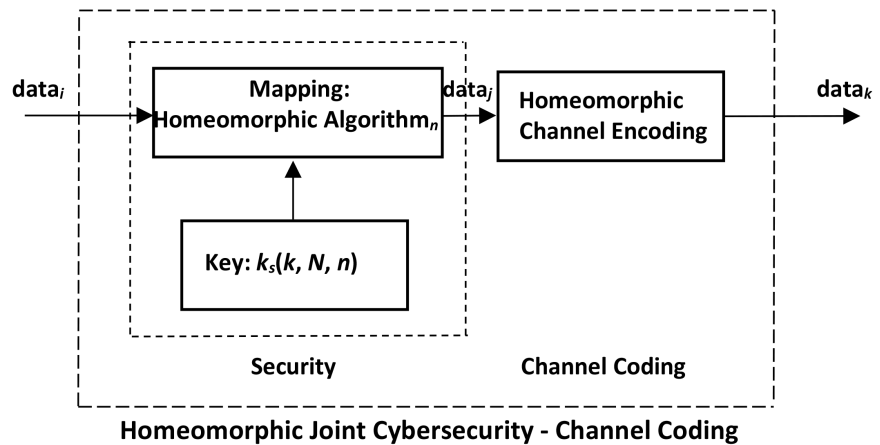
**Figure 10** shows the structural block diagram at the sender side for the introduced joint cybersecurity-channel coding using the implementation of the utilized logic homeomorphism. This is done by encrypting the incoming data<sub>*i*</sub> using homeomorphic mapping into data<sub>*j*</sub>. Then, the encrypted data<sub>*j*</sub> is fed into the next stage of channel coding (e.g., the MIMO encoder in **Figure 7**) to produce the corresponding homeomorphic channel encoded data<sub>*k*</sub>.

The first stage of homeomorphic mapping in **Figure 10** is done using specific homeomorphic Algorithm<sub>*n*</sub> which is produced by shared symmetric key  $k_s$  between sender and receiver since at the receiver data<sub>*j*</sub> can only be decrypted into

data<sub>j</sub> iff the same symmetric key  $k_s$  that generates Algorithm<sub>n</sub> is used, otherwise data<sub>j</sub> cannot be decrypted into the correct data<sub>i</sub>. The block diagram presented in **Figure 10** is a stage which is performed at the transmitter side as shown in **Figure 9(b)**, where the reverse-order and inverse-function of this stage is performed at the receiver side using firstly homeomorphic channel decoding (e.g., using the HV decoder from Section 4) to produce data<sub>j</sub> from received (erroneous) data<sub>k</sub> and then using symmetric key  $k_s$  to generate the corresponding homeomorphic Algorithm<sub>n</sub> to decrypt data<sub>j</sub> into the original data<sub>i</sub>.



**Figure 9.** Stages of wireless data transmission: (a) processing stages from data source to signal transmission where data is originated from a data source, filtering includes removing noise, signal processing provides signal enhancements, source coding provides data compression, security uses methods such as data encryption, channel coding is used for error detection and correction, signal coding includes conversion of signal form, spread spectrum includes bandwidth spreading, multiplexing is used for transmission from multiple sources over a single medium and networking consists of device networking over a network architecture, and antenna is used for the interfacing into the utilized transmission medium, and (b) the new scheme which is introduced for the first time in this article that joins the two stages of security and channel coding utilizing the implementation of logic homeomorphism.



**Figure 10.** The structural block diagram of the new scheme of homeomorphic joint cybersecurity-channel coding at the sender side, where the reverse-order and inverse-function of the inner block components are implemented at the receiver side.

The parameters for the symmetric key  $k_s$  are  $k_s = k_s(k, N, n)$  where  $k$  is the number of batched inputs,  $N$  is the number of outputs in the bijective mapping, and  $n$  is the index of the used logically homeomorphic Algorithm <sub>$n$</sub> , where  $N \geq k$ . The size of the key  $|k_s|$  is given by  $|k_s| = |k| + |N| + |n|$ . The key  $k_s(k, N, n)$  is generated using a random number generator (RNG), whether pseudo random number generator (PRNG) using mathematical formulas and a seed or true random number generator (TRNG) using unpredictable real-world events like atmospheric noise or thermal heat, combined with a mathematical algorithm. The space from which the symmetric key  $k_s$  can be chosen is vast depending on the combination of random choices via RNG for the symmetric key parameters  $\{k, N, n\}$ . The distribution of the shared symmetric key  $k_s$  to the sender and receiver can be done using mechanisms such as 1) key distribution center (KDC), 2) cryptographic handshake or 3) more advanced highly-secured modern method of quantum key distribution (QKD). The corresponding table for the utilized homeomorphic algorithms is stored at the sender and receiver sides with index  $n$  indicating the address of each homeomorphic Algorithm <sub>$n$</sub>  in the lookup table. The stored homeomorphic algorithms can have  $n$  variants, depending on the procedure for geometrically achieving logical homeomorphism via value-based space-partitioning that leads to spatial partitions of unique values, where 1) several  $n$  procedures can be used to obtain the corresponding several  $n$  homeomorphic algorithms of Algorithm <sub>$n$</sub>  or 2) alternatively a universal homeomorphic procedure Algorithm <sub>$u$</sub>  is stored to produce homeomorphic mappings which can produce for specific internal set of parameters  $\mathbf{p}$  the corresponding specific Algorithm <sub>$n$</sub>  where  $n = f(\mathbf{p})$ .

For example, a homeomorphic algorithm called HBF.V2 that can produce several variants of bijective mapping is as follows. One can note that, even for this same algorithm HBF.V2, several sub-variants can be produced for steps 2 and 3 since two choices are available in each of these two steps; choosing the upper (half) cells to be of value “0” or “1” and choosing the other lower (half) of cells to be of opposite value of “1” or “0”.

---

#### Algorithm HBF.V2

---

- 1) To get  $(k, N)$  homeomorphism, add adequate number of ancillary outputs  $N$  for  $k$  input data batch where  $N \geq k$ . Allocate a new column for each ancillary output.
  - 2) To get the first ancillary output, allocate a constant  $C_j$  to upper (half) cells in the corresponding table column and lower (half) cells as constant  $C_{j \oplus 1}$  where  $j \in \{0, 1\}$ .
  - 3) **For** the next ancillary output, **If** non-homeomorphism still occurs, **Then** allocate for same output tuples values  $v_j$  which are ones to upper (half) cells and zeros to lower (half) cells, or vice versa. **Else** allocate a constant for the rest which is the one's complement of the previously allocated constant to that remainder.
  - 4) **Do** step 3 **While** map items are non-homeomorphic.
-

**Example 4.** For the symmetric key  $k_s = k_s(k, N, n)$ , let the randomly chosen parameters using RNG to be  $k_s = k_s(3, 3, 5)$ , for the sent bit stream  $data_i = \{1, 1, 1\}$ , for which logic homeomorphism implementation using Algorithm<sub>5</sub> HBF.V1 (*i.e.*, table location address  $n = 5$  for the used bijective mapping algorithm HBF.V1) from Section 4, produces the following homeomorphic set of data sequences  $data_j = \{m_1 = (101), m_2 = (001), m_3 = (011)\}$  as follows:

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 0 | 1 | 1 |

This homeomorphic table, that illustrates the corresponding bijective mapping, is stored at both sides of sender and receiver for the cybersecurity stage. At sender, this resulting encrypted message flows into the next stage of homeomorphic channel encoding for which the resulting encoded sequences  $data_k = \{c_1 = (1110001011), c_2 = (0000111011), c_3 = (0011010111)\}$  are generated in parallel, where at the receiver side, as was seen previously using the HV decoder, the resulting corresponding survivor paths can be produced which generate the original transmitted correct messages from the corresponding received (erroneous) messages after which decoded data is fed into the next stage of homeomorphic message decryption. Further, and as detailed previously in Section 4, the existence of the homeomorphism property using the HBF Algorithm adds additional information to the original data that can be further used to correct for errors that are usually uncorrectable, depending on factors such as error pattern and free distance, as was shown within Example 3.

As stated previously and for symmetric key  $k_s = k_s(k, N, n)$ , it is to be noted that other homeomorphic-producing Algorithm <sub>$m$</sub>  where  $n = 1, 2, 3, \dots$ , can be utilized to generate other homeomorphic sets of data sequences  $data_j = \{m_1, m_2, m_3\}$ . For example, and for the bit stream  $data_i = \{1, 1, 1\}$  from Example 4, **Figure 11** shows several possible homeomorphic sets of data sequences  $data_j = \{m_1, m_2, m_3\}$  out of all possible homeomorphic data sequences that can be produced, where logic homeomorphism can be generated for any message cardinality  $|m_i| = N$  and not only for cardinality  $|m_i| = 3$  as shown in **Figure 11**. This wide range of possibilities for generating the used symmetric key  $k_s = k_s(k, N, n)$  using RNG to choose for parameters  $\{k, N, n\}$  makes it nearly impossible for any hacker to know exactly which symmetric key has been used. Thus, and by utilizing the corresponding used symmetric key  $k_s = k_s(k, N, n)$  receiver can only extract the correct sent message by only knowing the original symmetric key  $k_s = k_s(k, N, n)$ , and therefore further higher security level is achieved using this new encryption method.

It is to be noted that one can use at sender a multi-stage cybersecurity by using several sequenced homeomorphic algorithms  $\{\text{Algorithm}_1, \text{Algorithm}_2, \dots, \text{Algorithm}_n\}$  that are key-generated by  $\{k_{s1}, k_{s2}, \dots, k_{sn}\}$  where *each row-wise output* from *each* Algorithm <sub>$i$</sub>  becomes an input to the next encrypting algorithm and thus a

parallel *encrypting algorithm tree* structure is generated to model the sequenced phases of consecutive encrypting algorithms, where the multi-stage decryption at receiver is done in reverse-order and inverse-function via parallel *decrypting algorithm tree* structure of  $\{\text{Algorithm}_m, \dots, \text{Algorithm}_2, \text{Algorithm}_1\}$  that are key-generated by  $\{k_{sm}, \dots, k_{s2}, k_{s1}\}$  since one uses the same shared symmetric keys for the generation of the corresponding homeomorphic algorithms. This new and very important notion of parallel *cryptographic algorithm tree* is illustrated in **Figure 12**.

Also, one may note that this new homeomorphism-based cybersecurity scheme can be implemented in sequence in addition to the previously implemented steps of other security methods, such as RSA and AES encryption, that were applied prior to the phase of this new homeomorphic joint cybersecurity-channel coding scheme, where this used implementation will give further enhancement of confidentiality for the transacted data.

|       |   |   |   |
|-------|---|---|---|
| $m_1$ | 1 | 0 | 0 |
| $m_2$ | 0 | 0 | 0 |
| $m_3$ | 0 | 1 | 0 |
| (a)   |   |   |   |
| $m_1$ | 1 | 0 | 0 |
| $m_2$ | 1 | 1 | 0 |
| $m_3$ | 1 | 1 | 1 |
| (b)   |   |   |   |
| $m_1$ | 1 | 1 | 1 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 1 | 0 | 1 |
| (c)   |   |   |   |
| $m_1$ | 0 | 1 | 1 |
| $m_2$ | 1 | 0 | 1 |
| $m_3$ | 1 | 1 | 0 |
| (d)   |   |   |   |
| $m_1$ | 0 | 0 | 1 |
| $m_2$ | 1 | 0 | 0 |
| $m_3$ | 0 | 1 | 0 |
| (e)   |   |   |   |
| $m_1$ | 0 | 0 | 0 |
| $m_2$ | 0 | 0 | 1 |
| $m_3$ | 1 | 1 | 1 |
| (f)   |   |   |   |

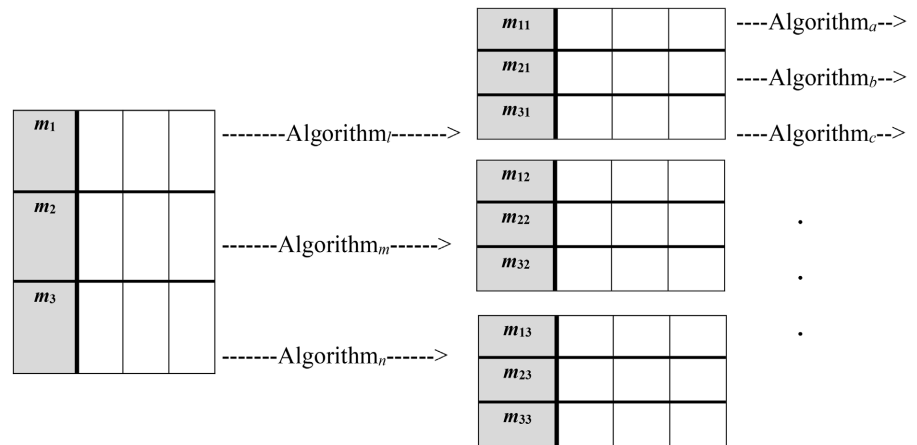
**Figure 11.** Different possible homeomorphic sets of data sequences  $\text{data}_i = \{m_1, m_2, m_3\}$  for the bit stream  $\text{data}_i = \{1, 1, 1\}$  by utilizing the corresponding key-generated homeomorphic  $\text{Algorithm}_m$ .

Further, since homeomorphic logic is information lossless and conservative logic is both information lossless and energy lossless, the homeomorphic mapping may implement further the constraint of logic conservativeness within which another variant of homeomorphic map can be produced which has the same number of zeros and ones in inputs and outputs and therefore energy preservation is achieved.

While several kinds of attacks can exist upon digital systems such as 1) software and data attacks (such as malware, ransomware, SQL injection, and zero-day exploits), 2) human and access attacks (such as phishing, man-in-the-middle, and insider threats) and 3) network and disruption attacks (such as denial of service, supply chain attacks, and DNS spoofing), it is to be noted that this newly introduced joint cybersecurity-channel coding design is intended to resist human and access attacks such as man-in-the-middle attacks to prevent eavesdropping (snooping) or secretly prying without permission into the transacted confidential data messages between two users. Further, for known-plaintext attack (KPA), which happens when an attacker has samples of both plaintext (unencrypted text) and ciphertext (encrypted text) and compares the pairs to discover the secret key, the more powerful chosen-plaintext attack (CPA), within which the attacker can pick their own plaintext and force the system to encrypt it to observe the resulting ciphertext and compares the pairs to discover the secret key, and chosen-ciphertext attack (CCA), where an attacker picks specific ciphertexts and get them decrypted

by the decryption system or tool (called an oracle) to use the results to discover the secret key, the introduced joint cybersecurity-channel coding scheme can further utilize key defense mechanisms such as lightweight authenticated encryption (AE) that combines confidentiality with strong message integrity codes (often called message authentication codes (MACs)) to ensure that any chosen ciphertext which is injected (by CPA adversary) is instantly detected and rejected.

It is also to be noted that the new scheme of homeomorphic joint cybersecurity-channel coding is different from homomorphic encryption, and one needs not to confuse between both of them. Homomorphic encryption (with main types of partially homomorphic encryption (PHE), somewhat homomorphic encryption (SHE), and fully homomorphic encryption (FHE)) is a special type of encryption where mathematical and logic tasks are run on locked secret data, where one does not need to unlock or decrypt the data first; when one finally unlocks the final result, it matches the answer one would get if one did the work on plaintext data. On the other hand, homeomorphism property is a bijective mapping which is used in this article for the new joint cybersecurity-channel coding scheme to achieve the objective of further confidentiality enhancement (as was fully demonstrated in this Section) and data integrity enhancement (thus better reliability) (as was illustrated in Section 4).



**Figure 12.** Cybersecurity multi-stage encryption by using several sequenced key-generated homeomorphic algorithms where each row-wise output from each Algorithm<sub>*a*</sub> becomes an input to the next encrypting algorithm and thus parallel encrypting algorithm tree structure is generated, where the multi-stage decryption at the receiver is done in exact reverse-order and inverse-function via parallel decrypting algorithm tree structure for the same utilized symmetric key-generated homeomorphic algorithms.

The new joint cybersecurity-channel coding method via logic homeomorphism implementation which is introduced in this Section will be further effectuated in the second part of this article using the implementation of regular lattice grids and the corresponding hierarchical effectuation utilizing nano-based carbon controlled-switching devices and methods.

## 6. Conclusions

A new joint cybersecurity-channel coding scheme utilizing logic homeomorphism is introduced for the first time in this article, and an important class of homeomorphic joint cybersecurity-Viterbi decoding is demonstrated. In addition to the demonstrated usefulness shown in this article for enhancing confidentiality within cybersecurity and improving data integrity (thus better reliability) within channel coding, logic homeomorphism is an essential property within low-power high-performance technologies for several advanced computing paradigms such as in nanoscale quantum computing. The second part of this article will present the important synthesis method of regular lattice grids that will be used for the corresponding ASIC hierarchical effectuation via nano-based carbon controlled-switching devices for the implementation of the newly introduced joint cybersecurity-channel coding design.

## Acknowledgements

The author acknowledges a granted sabbatical leave in 2024-2025 from The University of Jordan that enabled him to achieve this extended research.

## Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

## References

- [1] Al-Rabadi, A.N. (2004) Reversible Logic Synthesis: From Fundamentals to Quantum Computing. Springer.
- [2] Al-Rabadi, A.N. (2009) Reversible Viterbi Algorithm and Its Closed-System Q-Domain Circuit Design and Computation. *Journal of Circuits, Systems and Computers*, **18**, 1627-1649. <https://doi.org/10.1142/s0218126609005903>
- [3] Al-Rabadi, A.N. (2025) Convolutional Homeomorphic Error-Correcting Codes Utilizing Nano Controlled-Selector Architectural Effectuation. *Journal of Computer and Communications*, **13**, 187-226. <https://doi.org/10.4236/jcc.2025.1310010>
- [4] Bennett, C.H. (1973) Logical Reversibility of Computation. *IBM Journal of Research and Development*, **17**, 525-532. <https://doi.org/10.1147/rd.176.0525>
- [5] Landauer, R. (1961) Irreversibility and Heat Generation in the Computing Process. *IBM Journal of Research and Development*, **5**, 183-191. <https://doi.org/10.1147/rd.53.0183>
- [6] Nielsen, M. and Chuang, I. (2000) Quantum Computation and Quantum Information. Cambridge University Press.
- [7] Bishop, M. (2005) Introduction to Computer Security. Addison-Wesley.
- [8] Forouzan, B. (2008) Cryptography and Network Security. McGraw-Hill.
- [9] Stallings, W. (2017) Cryptography and Network Security: Principles and Practices. Pearson.
- [10] Proakis, J. and Salehi, M. (1994) Communication Systems Engineering. Prentice-Hall.
- [11] Lathi, B.P. (1995) Modern Digital and Analog Communication Systems. Oxford University Press.
- [12] Couch II, L.W. (2013) Digital and Analog Communication Systems. Pearson.