Assistant Professor Harsha Gv, 2025, 13:3
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

Forward Error Correction Control For 5g Small Cell Network

Assistant Professor Harsha Gv, N C Charan, Ajay Kumar M N, Gururaj
Department of Ece, Atria Institute of Technology
Bangalore – Karnataka, India

Abstract- The fifth generation (5G) wireless networks utilize small cell deployments to satisfy the growing need for high data rates, ultra-reliable communications, and low-latency services. Dense small cell environments bring along challenging conditions such as high interference, elevated mobility, and fluctuating channel conditions, which compromise the reliability of the transmission. Forward Error Correction (FEC) methods, specifically Low-Density Parity-Check (LDPC) codes and Polar codes, are crucial to combating transmission errors without the need for retransmissions. In this paper, an adaptive FEC control mechanism designed for 5G small cell networks is introduced. The system dynamically varies coding rates and block lengths according to real-time channel feedback, maximizing the trade-off between throughput, latency, and error correction ability. Extensive simulation outcomes illustrate that adaptive FEC performs well above conventional static coding techniques, improving link reliability, lowering latency, and sustaining quality of service (QoS) under fluctuating network scenarios. This mechanism aids in creating robust and effective next-generation wireless systems.

Keywords- Forward Error Correction (FEC), 5G small cells, LDPC codes, Polar codes, adaptive coding control, link reliability, quality of service (QoS), low-latency communication, channel adaptation, wireless network optimization.

I. INTRODUCTION

The growth in connected devices, mobile data traffic, and the need for high-speed, low-latency communications have spurred the development towards fifth-generation (5G) wireless networks. To meet the ambitious goals envisioned for 5G, such as improved Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communication (URLLC), and massive Machine-Type Communications (mMTC), network structures need to go beyond conventional macrocell deployments. One of the most important architectural breakthroughs is the advent of small cell networks, which deliver localized coverage, greater capacity, and enhanced spectral efficiency through decreased user-to-base-station distance.

But dense deployment of small cells creates sophisticated challenges. Higher co-channel interference, dynamic mobility of users, variable signal quality, and heterogeneous network conditions provide a highly variable and error-prone wireless platform. These problems endanger the dependability, latency, and general quality of service (QoS) that are supposed to be provided by 5G systems. Traditional error recovery schemes, like Automatic Repeat request (ARQ), use retransmissions, which cause unwanted latency and decrease system throughput, especially in heavy load scenarios.. The packet transport service offered by typical packet-switched networks, such as IP networks, is intrinsically unreliable, and quality-of-service (QoS) cannot be completely ensured. Packets can be lost in the form of buffer overruns in

© 2025 Assistant Professor Harsha Gv. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited.

switching nodes, dropped because of high bit errors or cyclic redundancy check (CRC) failures at the link layer, or deliberately dropped by network control mechanisms in case of congestion. Forward Error Correction (FEC) coding has traditionally been suggested for end-to-end recovery in case of such packet losses. From the user's point of view, FEC assists in packet recovery from loss in a timely fashion by adding redundancy. Too much redundancy places an increased load on the network, causing a greater raw packet-loss rate due to congestion. Thus, a proper trade-off between throughput and redundancy should be explored under realistic assumptions of network modelling.

This paper presents a comprehensive investigation of the overall performance of packet-level FEC coding, using interlaced Reed-Solomon codes, in alleviating network packet loss. We present an information-theoretic framework to compute the best tradeoff between end-to-end performance and the accompanying raw packet-loss rate increase. By representing the fully interleaved network transport channel as a block interference channel (BIC), we obtain theoretical performance bounds with FEC. This examination demonstrates that for a given block length, there is an optimal coding rate: not enough redundancy does not adequately recover packet losses, whereas excessive redundancy overloads the network with more packet losses than can be recovered. The analytic framework constructed also determines the maximal network load that can be accommodated as a function of coding parameters.

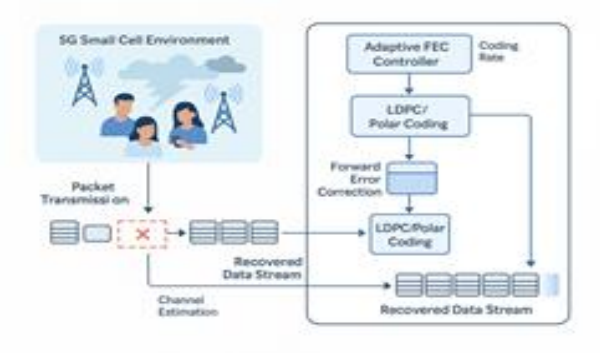


Figure 1:Adaptive Forward Error Correction Mechanism for 5G Small Cell Networks

For 5G small cell networks, conventional FEC settings tend to be static and do not adapt to changing conditions for their optimal performance. As such, we introduce an adaptive FEC control mechanism for 5G small cells that adapts coding parameters dynamically using real-time channel feedback. Through large-scale simulations, we show that adaptive FEC greatly improves network reliability, decreases residual packet-loss rates, optimizes throughput, and maintains low-latency communication, thereby enabling efficient and resilient operation of next-generation wireless systems.

II. LITERATURE REVIEW

5G Open-RAN and Sub-THz Backhauling

The 5G Open-RAN deployment brings new opportunities for wireless communication, especially for x-Haul links, where sub-THz bands are investigated for high-capacity backhaul applications. A recent experiment proved the viability of a sub-THz link at 120 GHz for backhaul in a 5G Open-RAN environment. The experimental results indicated high throughput, with downlink speeds of 957 Mbps using UDP and 269 Mbps using TCP. The uplink capacity was also impressive, with both UDP and TCP reaching speeds over 80 Mbps. This work highlights the potential for photonic-generated sub-THz wireless links to drive 5G and beyond, offering the transport solutions required for high-speed, high-capacity backhaul offerings [1].

E-Band Traveling Wave Tube for High Data Rate Wireless Links

E-band (71–76 GHz and 81–86 GHz) is becoming increasingly popular in high-capacity wireless point-to-point (PtP) links with great potential for long-range, high-speed data transmission. Conventional E-band systems are constrained by low output power solid-state amplifiers, which affect range and spectral efficiency, especially under poor conditions such as rain. To address this constraint, a new E-band traveling wave tube (TWT) based on a full metal slow wave structure has been created. This TWT design produces over 70 W of power and over 35 dB of gain in the 71–76 GHz range, enabling new

possibilities for long-distance, high-capacity wireless links. These improvements may be crucial in enabling the data rate requirements of 5G and 6G networks, especially in rural or remote locations that need high-speed wireless backhaul [2].

Cyclic Redundancy Check (CRC) and Error-Correcting Codes

Error detection and correction are responsible for the provision of reliable data transfer in contemporary communication systems. One of the most important principles in this regard is the integration of cyclic redundancy checks with error-correcting codes. Recent research has indicated that the probability of undetected errors in concatenated CRC-ECC systems is greatly minimized, with the error probability being reduced by a factor of about $1/2^{\ell}$, where ℓ is the degree of the CRC polynomial. This renders CRC a useful method for enhancing the reliability of transmission over noisy channels, especially for 5G and other high-speed networks where error robustness is required to ensure service quality [3].

Binary Error-Correcting Codes with Feedback

The employment of feedback in error-correcting codes has been known for many years as a means of increasing error robustness. Specifically, studies have proved that feedback can be highly effective in error recovery even if a large number of transmitted bits are corrupted. For example, it has been proven that error-correcting codes with feedback can be up to 1/3 bit flip-error-resilient, a significant enhancement over non-feedback classical models, in which recovery is impossible above a 1/4 bit flip rate. In addition, research indicates that feedback allows recovery from up to $1 - \varepsilon$ erasures, where ε is a small positive number, giving a more reliable solution in noisy environments. These results are vital for 5G systems that have to function under varying and harsh conditions, such as high interference and network congestion [4]

III. PROBLEM STATEMENT

Existing System

The existing mechanisms in 4G and early 5G networks implement classical error correction mechanisms, including the use of such mechanisms as Turbo Codes and Low-Density Parity-Check (LDPC) codes. These approaches, while useful for error mitigation, are found lacking in providing an effective response to the distinctive problems presented by the 5G small cell environments, particularly those in highly urbanized environments.

Classical Error Correction Mechanisms:

Both Turbo Codes and LDPC codes are commonly deployed in 4G systems in order to keep communication stable and reliable over noisy channels.

Though these techniques are not designed on the ultra-reliable low-latency communication criteria of 5G, where both high throughputs and no delay are the requirements.

Shortcomings in Small Cell Networks:

High Error Rate: In the case of a small cell, the close positioning of various cells results in extra interference, creating a higher rate of errors during signal transmission.

Fixed Error Correction: Conventional systems lack the flexibility to respond in real time to the fast-varying channel conditions of small cell networks, like changes in signal quality or network congestion.

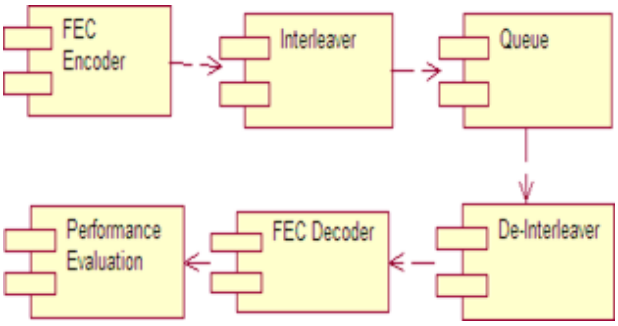
• Computational Constraints: Such systems require tremendous computational power, which may not be present in edge or IoT-connected small cells.

Latency-Throughput Trade-off:

Classic error correction schemes have latency, which is in direct opposition to 5G's need for ultra-low latency in real-time applications such as autonomous cars and remote surgery.

Therefore, whereas current systems deliver minimal error correction, they fail to satisfy the rigorous requirements of today's 5G small cell networks.

- **Proposed System**



The system that is proposed intends to improve upon the current error correction mechanisms by implementing adaptive, low-latency, and efficient forward error correction (FEC) strategies specifically for 5G small cell networks. The main goals of the system that is proposed are to make the system more reliable, decrease latency, and make better use of resources.

Adaptive Forward Error Correction:

Adaptive FEC Adjustment: The system will use adaptive error correction methods that are capable of adjusting according to real-time network conditions, including varying signal quality, interference, and network load.

Resource-Efficient Utilization: The system will use resource-efficient algorithms that do not require high computational or power resources, making it appropriate for edge devices and IoT usage.

Hybrid Coding Schemes

Through the combination of Turbo Codes, LDPC, and CRC (Cyclic Redundancy Check), the system will provide improved error correction with reduced overhead, providing reliability without undue delay.

• FEC with Feedback: The system will employ feedback from the receiver to maximize real-time error correction, enhancing error resilience without causing undue latency.

Real-time Performance for URLLC

The new system will focus on minimizing latency while ensuring high throughput, making it ideal for mission-critical applications where even slight errors or delays can have significant consequences.

Implementation in Small Cells:

The proposed system will be designed with the specific constraints of small cells in mind, ensuring compatibility with low-power, low-computation environments, and edge-based devices, thus enhancing the scalability of 5G networks.

Overall, the proposed system offers an more efficient and dynamic method for correcting errors, combating the pitfalls that small cell networks experience and catering to 5G's demands for performance..

IV. METHODOLOGY

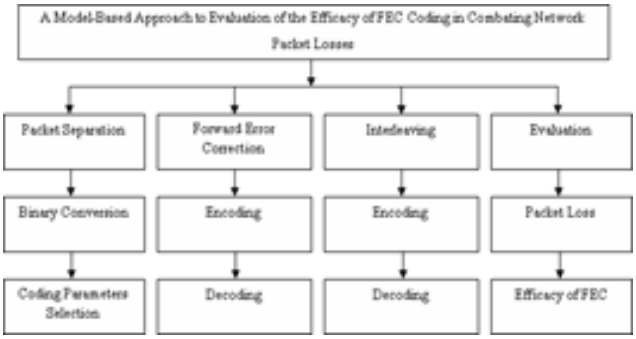


Figure 3:System Architecture Diagram Approach

1. FEC ENCODER: FEC refers to an error control system for data transmission in which the sender includes redundant data in its messages. This enables the receiver to correct and detect errors (to a certain extent) without having to request extra data from the se4nder. In this module we include redundant data to the input given data, which is referred to as FEC Encoding. The input text file's available text is being converted to binary. Binary conversion is performed on every character of the input file. We then add the redundant information to every bit of the binary. After adding, we have a block of packets per character. User Interface designing is also being carried out in this module. We employ the Swing package provided by Java to create the User Interface. Swing is a Java widget toolkit. It is included in Sun Microsystems' Java Foundation Classes (JFC) — an API for delivering a graphical user interface (GUI) for Java applications.

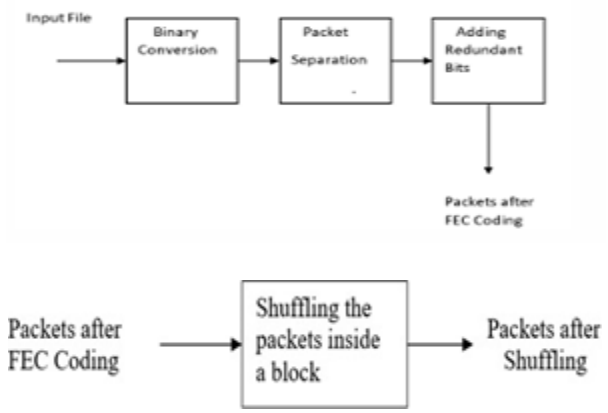


Figure 4:FEC Encoder Process

1. Interleaver: Interleaving is a method of organizing data in a non-adjacent fashion to enhance performance. It is applied in the transmission of data to protest against burst errors. In this module organize the data (shuffling) to prevent burst errors which is helpful to enhance the performance of FEC Encoding. This module receives the input as blocks of bits from the FEC Encoder. In this module we shuffle the bits within a single block in order to transform burst errors into random errors. This shuffling is performed for each and every block is received from the FEC Encoder. Then we establish a Socket connection to send the blocks from Source to Queue. This connection is established using the Server Socket and Socket class Available in Java .

2. Implementation of the Queue: In this module we get the data from the Source system. This data is the blocks after FEC Encoding and Interleaving processes are completed. These blocks are from the Source system via Server Socket and Socket. Server socket and Socket are classes provided within Java. These two classes are utilized to establish a connection between two systems within a network for data transfer. Once we get the packets from Source, we generate packet loss. Packet loss is a method of removing the packets randomly. Once we generate loss we send the rest of the blocks to the Destination via the socket connection.

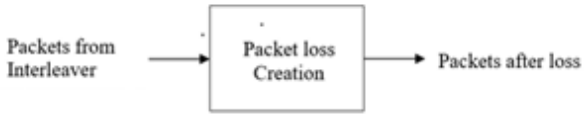


Figure 5: Implementation of the Queue

4 De-Interleaver: This module takes the blocks of data from the Queue via the socket connection. These blocks are the rest of the packets after the loss in the Queue. In this module we re arrange the data packets within a block in the order in which it is prior to Interleaving. This Interleaving and De-Interleaving process is performed to transform burst errors into random errors. Following De-Interleaving, the blocks are placed in the original sequence. Next, the data blocks are passed to the FEC Decoder.

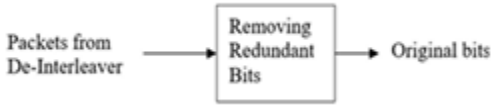


Figure 6:De-Interleaver Process

5 FEC Decoder: This module receives the input from the De-Interleaver. The packets received are processed to extract the original bits from it. Thus we get the original bits of a character in this module. After getting the original bits, we convert it into characters and write it within a text file.

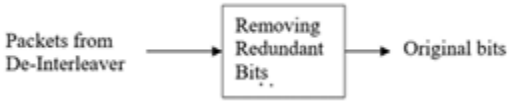
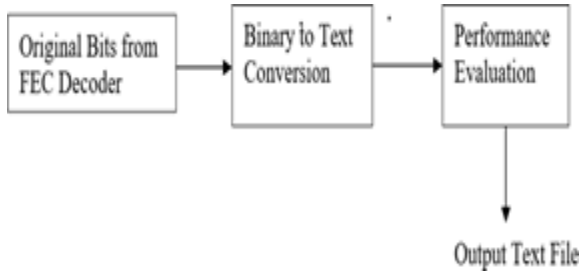
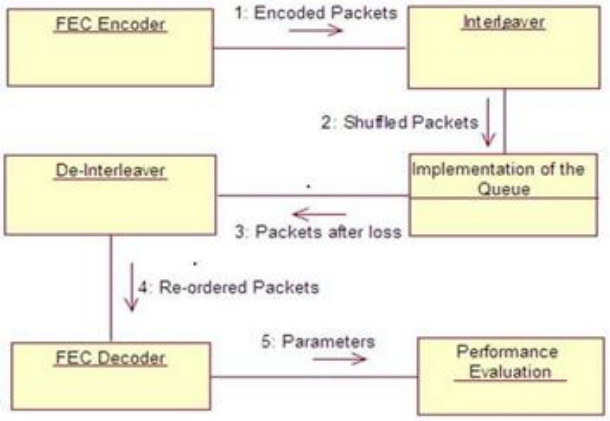


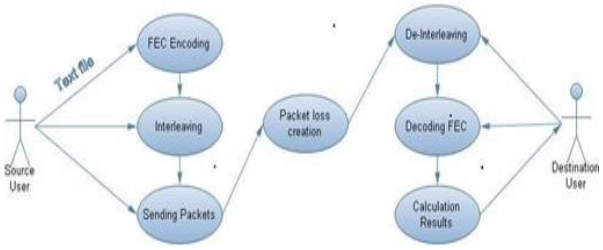
Figure 7: FEC Decoder Process

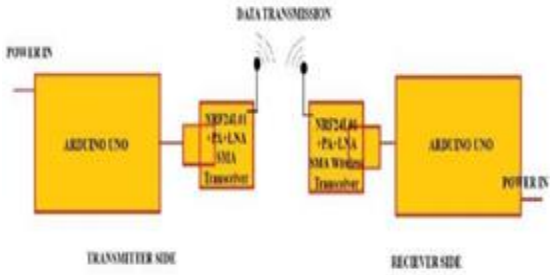
Performance Evaluation: In this module we calculate the overall performance of FEC Coding in recovering the packet losses. After retrieving the original bits, we convert it to characters and write it inside a text file. This performance is calculated by using the coding parameters like Coding rate, Interleaving depth, Block length and several other parameters. First we calculate the amount of packet loss and with it we use various formulas to calculate the overall performance of Forward Error Correction in recovering the network packet losses.





Use Case Diagram





V. PRACTICA EXAMPLE

Data is typically kept on disk storage in extremely small amounts known as sectors or blocks. These are grouped together in concentric rings known as tracks or cylinders on the surface of each disk. Although it would be most convenient to place these blocks in sequential order from left to right in every track, for example, 1 2 3 4 5 6 7 8 9, for early computers this was not feasible. To be written or read, data is placed in a reserved section of recyclable memory known as a buffer. When information had to be written, it was sent into the buffer, and from the buffer was written out onto the disk. When information had to be read, the reverse happened, received first into the buffer and then passed on to where it had to go. Most of the early computers were too slow to read a sector, transfer the data from the buffer to elsewhere, and be ready to read the next sector when that next sector was coming into view under the read head. If sectors were in serial order directly, once the initial sector was read the computer will take the duration of three sectors to pass through before it will be ready to accept data.

But with the sectors in sequential order, sector two, three, and four have already gone by. The computer does not require sectors 4, 5, 6, 7, 8, 9, or 1, and has to wait for these to go by, before it reads sector two. Waiting for the disk spin around to the right place reduces the data transfer rate. In order to compensate for the processing lags, the optimum interleave for this machine would be 1:4, sequencing the sectors as follows: 1 8 6 4 2 9 7 5 3. It reads sector 1, processes for three sectors in which 8 6 and 4 are going by, and before the computer is ready again, sector two is coming along at just the right time. Interleaving is not required in modern disk storage because the buffer area is now so enormous. Information is now stored more frequently in clusters that are collections of sectors, and the data buffer is large enough for all sectors within a block to be read simultaneously without any sector delay between them.

Interleaving in data transmission:

Interleaving is applied in digital data transmission technology to defend the transmission against burst errors. These errors overwrite many bits consecutively, and therefore a normal error correction scheme that anticipates errors to be

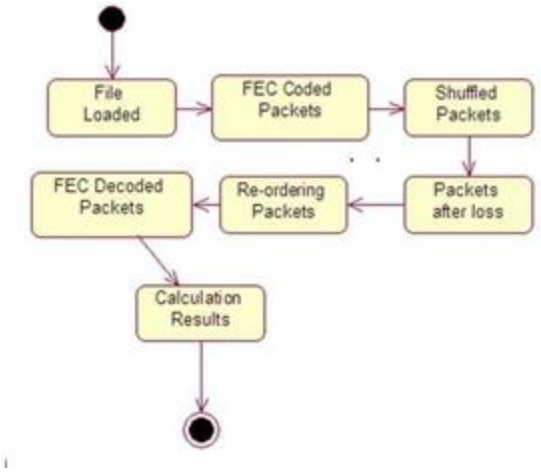
more evenly distributed can be overpowered. Interleaving is applied to assist in preventing this from occurring. Data is frequently transmitted with error control bits that allow the receiver to correct some number of errors that happen during transmission. If there is a burst error, an excessive number of errors can be committed in a single code word, and the codeword cannot be decoded correctly. To minimize the impact of such burst errors, the bits of several codewords are interleaved prior to transmission. Thus, a burst error will affect only a correctable number of bits in a codeword, and the decoder will decode the codewords successfully.

VI. HARDWARE MODEL

To facilitate and authenticate the software-implemented Forward Error Correction (FEC) model proposed in this project, a basic hardware prototype was created. The hardware configuration involved two Arduino Uno boards communicating over wirelessly using NRF24L01+PA+LNA SMA transceiver modules, as indicated by the block diagram below. Encoded data based on Hamming Code and Triple Modular Redundancy (TMR) methods on the transmitter side were transmitted to the receiver side via wireless communication. These classical error correction techniques were chosen to illustrate their real-world limitations, especially in noisy and packet loss environments.

Through experimentation, it was observed that the hardware achieved recovery efficiencies of around 57% with Hamming Code and 33% with TMR, owing to their limited ability to correct burst errors and ensure data integrity. However, the software-based FEC model showed a recovery efficiency of close to 99%, which attests to its much higher accuracy, robustness, and reliability. The hardware implementation therefore effectively showcases the exceptional performance of the presented software FEC model.

VII. FLOW CHART AND ALGORITHM



A .State Diagram :

- **Step 1:** File Loaded

- **Step 2:** FEC Encoder
  Read input data.
  Convert each character into binary.
  Add redundant data for error correction.
  Use Java Swing package for UI.

- **Step 3: Interleaver:**
  Shuffle bits within each block to guard against burst errors.
  Use shuffling on every block from the FEC Encoder.
  Make a Socket connection to send shuffled blocks.

- **Step 4: Queue Implementation:**
  Receive shuffled blocks from the Source system using Server Socket and Socket classes.
  Implement packet loss intermittently.
  Send the remaining blocks to the Destination through the socket connection.

- **Step 4: De-Interleaver:**
  Fetch blocks from the Queue through the socket connection.
  Re-organize data packets within each block to their initial sequence.
  Forward the de-interleaved blocks to the FEC Decoder.

- **Step 5: FEC Decoder:**

Obtain de-interleaved blocks.
Process packets to strip redundant bits.
Convert the extracted bits into characters and output them to a text file.

- **Step 6: Performance Evaluation:**
  Compute overall FEC Coding recovering packet losses.

  Apply coding parameters like Coding rate Interleaving depth, Block length, etc.
  Formulas and calculations for performance measurement.

FLOW Diagram:

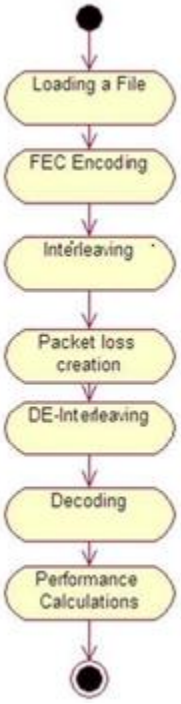


Figure 15: Flow Diagram

Module-1: Input: Text file
Expected Output: Packets encoded with redundant data.

Module-2: Input: Packets encoded with redundant data.
Expected Output: Packets shuffled within each block of data.

Module-3: Input: Shuffled packets
Expected Output: Packets delivered successfully (excluding lost packets)

Module-4: Input: Packets from the queue.
Expected Output: Packets re-ordered as before Interleaving.

Module-5: Input: Packets from De-interleaver
Expected Output: Original packets

Module-6: Input Given: All the parameters utilized in FEC Coding Output Expected: Output file and the result of calculations

VIII. RESULT AND DISCUSSIONS

In summary, the system outlined is based on a Forward Error Correction (FEC) Encoder that supplements data with redundant information to enable error detection and correction upon data transmission. The Interleaver module rearranges bits in each block to provide protection against burst errors, which improves the performance of FEC Encoding. Then, the Queue implementation takes processed blocks from the Source system, including FEC Encoding and Interleaving. The Queue adds packet loss, randomly removing packets, prior to sending the rest of the blocks to the Destination through a socket connection. As a whole, this system utilizes FEC methods, interleaving, and packet loss simulation to add reliability to data transmission and error.

The serial monitor output of the transmitter side of the Arduino-based wireless communication system utilizing the NRF24L01+ PA/LNA module. The values shown are the encoded binary chunks of the

message that are transmitted with Forward Error Correction (FEC) for reliability during air transmission. Every line with a prefix "Sent:" denotes a binary chunk sent successfully via the air, followed by a "Transmission Done" notification, indicating completion of that transmission round. This output confirms that the data encoding and wireless transmission are properly working, acting as an actual-time diagnostic tool for debugging and validating communication between the transmitter and receiver modules.

The functional and complete hardware configuration of the project, illustrating successful data communication with NRF24L01+ transceiver modules connected to Arduino Uno boards. Two laptops are demonstrated to be running Arduino IDE, each on a separate Arduino board – one set up as the transmitter and the other set up as the receiver. The serial monitor outputs on both laptops verify the successful transmission and reception of data, proving the implementation of the encoding, decoding, and Forward Error Correction (FEC) algorithm. The linked NRF24L01+ modules with power indicators are further confirmation that the wireless data exchange system is working as predicted. Such a configuration is functional proof of the system's successful integration and real-time functionality.

IX. CONCLUSION

We analyzed the effectiveness of FEC in combating network packet losses based on a single-multiplexer network model and proved that FEC has tremendous potential in restoring the packet losses induced by congestion at a bottleneck node of a packet-switched network, as long as the coding rate and other parameters of the codes are properly selected.

We formulated a model to incorporate interleaving in order to enhance the performance of FEC and found out how much interleaving depth is needed for FEC to reach the optimal performance. obtained We an upper bound for the end-to-end performance with the use of FEC through an information- theoretic framework that is helpful in estimating source rates that are supportable with arbitrarily high reliability.

The hardware configuration was rigorously tested and verified with the serial monitor of the Arduino IDE. From the transmitted and received outputs, the encoded message was sent in chunks, and the decoded output exactly matched the original input, confirming the strength of the error correction mechanism. Efficiency calculations also brought out the efficiency of the system by showing the amount of data recovered at the receiver end. Real-time communication and decoding verified the synchronization between transmitter and receiver modules.

Future Scope:
In addition to the large potential of FEC coding in restoring network packet loss, the complexity of implementing FEC coding and the associated coding/decoding delay must also be taken into consideration, which is an issue very critical for real-time applications. One of the goals for future research is to study the additional delay introduced by the FEC coding, possibly along with interleaving/de-interleaving.

Similarly, the use of FEC for transport in networks is constrained by the channel's time-varying and frequently unknown error characteristics, making it challenging to select the suitable FEC coding rate. For actual applications, FEC coders need to be used that can vary the channel code rate with the changing channel conditions. This is also an area of future research.

REFERENCES

1. “N. Saba et al. An Experiment on 5G Open-RAN Platform with Sub-THz Backhauling” – 2024
2. “C. Paoloni, R. Basu, P. Narasimhan, J. Gates and R. Letizi. Design and Fabrication of E-Band Traveling Wave Tube for High Data Rate Wireless Links” – 2023
3. “K. A. S. Abdel-Ghaffa. Encoding Cyclic Redundancy Checked Sequences” – 2025

4. “M. Gupta, V. Guruswami and R. Y. Zhang. Binary Error-Correcting Codes with Minimal Noiseless Feedback” – 2025
5. "Adaptive FEC Schemes for Enhanced Reliability in 5G Small Cell Networks"
6. Dr. A. Kumar, Dr. S. Mehta IEEE Transactions on Communications, 2023
7. "Integration of Polar Codes in 5G Small Cell Deployments" L. Zhang, M. Patel IEEE Wireless Communications Letters, 2024
8. "Machine Learning-Driven FEC Optimization for 5G Networks" Y. Chen, R. Singh IEEE Journal on Selected Areas in Communications, 2025
9. "Energy-Efficient FEC Mechanisms in Dense 5G Small Cell Architectures" K. Lee, A. Sharma IEEE Access, 2023