EAI Endorsed Transactions
on Energy Web
Research Article EAI.EU

ACRCP -Protocol for Delay Tolerant Networks for Effective Green Computing Environment

V. Preetha[1], K. Balasubramanian[2]

[1]Assistant Professor, Department of Computer Science, Sri. S. Ramasamy Naidu memorial College, Sattur, Virudhunagar, Tamil Nadu, India
[2]Professor, Department of Computer Applications, Kalasalingam Academy of Research and Education (Deemed to be University), Krishnankoil, Tamil Nadu, India

Abstract

There is no universally accepted definition of a green building. Generally, it refers to an ideal living environment created with respect for ecological principles and environmental considerations, combined with the principle of ecological design. Green buildings not only improve living conditions and energy efficiency but also contribute to urban air quality, enhance landscapes, and positively impact the physiological and psychological well-being of individuals. To achieve the goals of green building, it is imperative to have convenient and accurate selection of plant materials, create a sustainable green system, and establish a comprehensive green building plant resource information system.

Over the past few decades, numerous quantitative studies on the ecological functions of green plants have been conducted worldwide. Various test methods have been employed under different green conditions to quantify these functions and obtain numerous quantitative values. Concepts such as green area, green coverage, leaf area index, leaf area, green rate, vertical green coverage area, and overall green quantity have been introduced. Delay Tolerant Network (DTN) engineering faces challenges in ensuring end-to-end availability in cloud computing. Consequently, transporting data over such networks is challenging since most Internet applications rely on persistent end-to-end connections. Since the standardization of the DTN architecture, significant research has been dedicated to developing routing protocols for various mobility scenarios, aiming to improve segment delay, delivery, and data delivery ratios.

Keywords: Delay Tolerant Network, Green computing, Routing Protocol

Received on 02 11 2024, accepted on 30 05 2024, published on 28 06 2024

doi: 10.4108/eetiot.6455

1. Introduction

Delay Tolerant Networks (DTNs) play a vital role in versatile remote hubs prepared to transmit information when they are within transmission range. Due to the infrequent and often disconnected nature of DTNs, these hubs frequently experience network interruptions. Standard ad-hoc routing protocols are ineffective under such conditions. Therefore, for routing in these networks, data transmission requires the use of replication, erasure codes, and multipath techniques, as discussed in [1, 2], to improve both delivery ratio and delivery delay. However, these methods do not consider the use of an acknowledgment path or an inoculation strategy [3], which can lead to scalability issues. One approach is to consider the availability of an acknowledgment path [4, 5].

The Delay Tolerant Networking architecture supports a custody transfer concept implemented through a recognized exchange of a determined, reliable buffer [6]

A node "taking custody" of a segment commits to deliver the segment to its destination or another Node

Handler node, effectively extending the end-to-end transmission [7] to new areas of the cloud. The goal of custody transfer is to use hop-by-hop reliability to enhance end-to-end reliability and to free retransmission buffers at the source as soon as possible. To achieve this, the node taking custody must generally reserve buffer space for segments it takes custody of, resulting in reduced buffer availability for taking custody of subsequent segments or for its regular task of forwarding segments. When faced with constant demand, a Node Handler unable to release or transfer custody of its segments will eventually exhaust its buffer resources, leading to a form of DTN congestion [8]. This congestion can easily cause head-of-line blocking, preventing further data flow even when some links are available. Managing congestion at a Node Handler is a complex task.

In this paper, we address the issue using the latter two options, leaving the first option for future work. After many years of empirical research in various aspects of Delay Tolerant Networking, such as routing, transport protocols, and layer integration, DTN technology has achieved significant development. The creation of a reliable set of operational plans and related standards under the guidance of the Consultative Committee for Space Data Systems (CCSDS) and the Internet Research Task Force (IRTF) has been crucial to this progress [10].

Power's DTN exploration collective has significantly enhanced the practical implementation of DTN models, establishing them as viable solutions for global internetworking [11,12]. Building on this development, several studies [1, 16, 17] have highlighted the benefits of DTN systems and strongly advocate their use in challenging environments through the Bundle Protocol [16], which encapsulates most functionalities required by an overlay network. Our work addresses a crucial area that has not yet received extensive attention despite its potential importance: data segmentation over DTNs.

TCP or even less aggressive protocols that simply utilize the remaining capacity, as has been suggested for the foundational operation of distributed applications [13]. Despite error-covering mechanisms built directly into the codecs, simple redundancy or FEC schemes [14] can be added to compensate for losses with minimal overhead. This non-adaptive approach works fine as long as users have a stable end-to-end path—such as a sufficiently stable access link to the network—so occasional (congestion) losses remain the primary cause of perceived degradation in path capacity or latency. However, this does not hold true for mobile nodes, where interference, attenuation, or coverage gaps can affect path qualities more severely and unpredictably. In such cases, the physical network can quickly become the bottleneck even for low-bitrate audio, leading to significant instantaneous loss rates or considerable delay. This, in turn, results in losses when packets are dropped on the receiver side because they miss their playout deadline. When such situations occur, simply reducing the transmission rate of individual audio streams, even if implemented, would not help. Different mechanisms for adaptation are required instead [18].

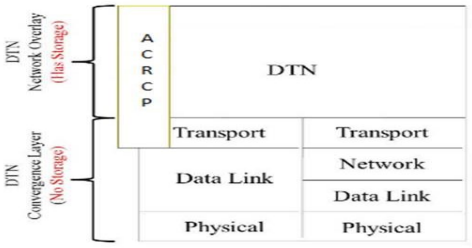

Fig 1. ACRCP protocol in TCP/IP suit

In this unique context, we propose the Adaptive Coding Scheme for Reliable Communication Protocol (ACRCP) as a practical approach that addresses many of the networking challenges related to fragmenting over DTNs. Positioned above the transport layer in TCP/IP, as shown in Fig. 1, ACRCP enables "fragmented" information to be transmitted through DTN "segments." This supports seamless fragment processing with minimal latency while ensuring reliable delivery, facilitating real-time "playback" review of received information. Potential applications that could benefit from this framework include one-way voice, video, or continuous telemetry fragmenting.

ACRCP was designed with the Interplanetary Internet (IPN) [15-17] and its associated issues in mind. Despite its initial focus, our proposal could also be applicable in terrestrial delay/disruption-tolerant environments where network nodes either follow a predetermined path or move freely within a dynamic topology, experiencing intermittent disruptions as they move beyond communication range. Such networks exhibit characteristics similar to space internetworks regarding bandwidth limitations and connection availability, and thus, could be adapted by a common architectural framework.

2. Related Works

Nonetheless, to the best of our knowledge, the issue of information spilling with respect to DTNs hasn't been extensively considered. In [7], destruction coding techniques were used to create an interference-tolerant video stream. This method ensures that in the event of interruptions, continuous video content is still provided to users by embedding additional "summary frames" into the main stream. In [8], the authors build on [14] by proposing a robust streaming technique based on an on-the-fly coding scheme, where encoding and decoding processes are carried out at the source and destination nodes, respectively.
In [15], it was discussed that each connection can have varying sizes, necessitating re-encoding at intermediate nodes. This idea was further developed in [15], where the authors proposed an encoding strategy at intermediate cloud nodes to explore the potential of Forward Error Correction

schemes within the bundle protocol [10]. In [12], a Mobility Aware MAC Protocol for Providing Energy Efficiency and Stability in MWSNs is presented. This work maintains a connection status history while the MOX-MAC nodes transmit their data to the static nodes. If the connection quality deteriorates, the MOX-MAC will drop or ignore the packet.

3. Methods and Materials

3.1.1 Delay aware Relay Node Selection Algorithm

In this paper, a power and delay-aware relay node selection algorithm is proposed for cluster-based green computing. Initially, the broadcasted data packets are assigned priorities as follows: for collected data from its own node, the priority is 1; for collected packets from other nodes, the priority is 2. When each relay node is awake, the Cluster Head (CH) queries the relay to determine its reward value. Based on this reward value, the CH decides whether to forward its packet to the selected relay or wait for other relays to wake up. The goal of the forwarder is to choose a relay that minimizes a combination of waiting delay, reward, and probing cost [10]. The reward is calculated based on the combined parameters of channel gain [10] and Relative Energy Usage (REU) [11]. The CH ultimately selects the relay with the best reward and the least waiting time, considering minimal probing costs, and forwards the data packet through this chosen relay node.

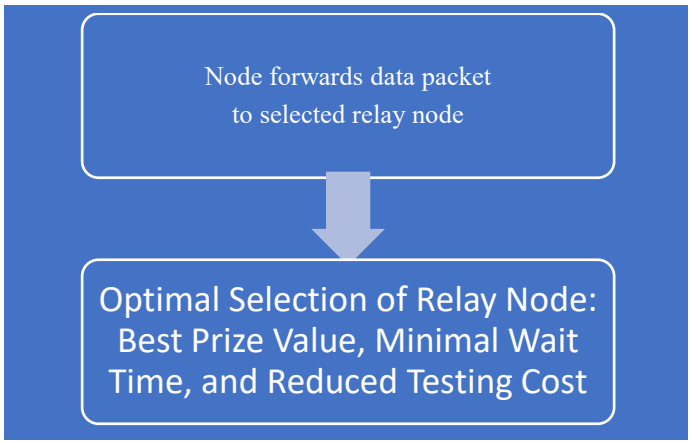

Fig 2. Block Diagram of Proposed Technique

When a node needs to forward a data packet, it selects a suitable relay node along the transmission path. This relay node selection is based on the power consumed and the delay involved in processing. The cluster head of the group selects the relay node by evaluating the reward value, waiting time, and probing cost. The reward value depends on the Relative Energy Usage (REU) and channel gain. The REU is defined as the ratio of the cost of generated and transmitted packets to the cost of generated packets [11]. The relay node with the best values is chosen for sending the data packet. This process is described in Algorithm 1.

3.1.2 Adaptive Coding Scheme for Reliable Communication Protocol (ACRCP)
Link Reliability Calculation:

ACRCP estimates the nearby available bandwidth by using idle channel time. Generally, a channel at a node can be either idle or busy. The channel is idle if the node is not in the following three states. First, the node is transmitting or receiving a packet. Second, the node receives an RTS or CTS message from another node, which reserves the channel for a specified period. Third, the node detects a busy carrier with a signal strength larger than a certain threshold, called the Carrier-Sensing Threshold, but cannot interpret the content of the message. By monitoring the amount of idle channel time (T_idle) during each period (T_p), the nearby available bandwidth can be estimated. In the context of a MAC layer that supports priority-based scheduling, such as IEEE 802.11e [12], estimating local available bandwidth may be more complex, as high-priority flows may be allowed to dominate the bandwidth over lower-priority flows. In such a network, the local available bandwidth to a stream may depend on its priority. A method for calculating the local available bandwidth for each priority is presented in our work [2] and can be easily integrated with ACRCP.

Congestion Control

Upon detecting congestion, DTN Node Handlers initiate a series of algorithms to determine which segments to relocate and identify alternative Node Handlers for the transfer. Subsequently, using the push operation defined in traditional DTN control protocols [13], the Node Handler transfers the selected segments to alternative buffer locations. When sufficient buffer resources are available, the Node Handler invokes a retrieval algorithm to pull segments it previously controlled towards their destinations.

Segment Selection

When an IP switch faces internal line disablement, it selects segments for transmission using the push control exchange operation. Typically, either an arriving or buffered segment can be chosen. Push decisions are categorized by factors such as urgency, size, and priority. The push strategy prioritizes segments currently in the buffer or newly arrived segments for transmission. This approach is akin to a drop strategy in that it selects a specific segment or packet for removal from the buffer,

but differs in the action taken. In a push strategy, segments can be evicted from the buffer, though this action is typically chosen only when no available buffer exists at any adjacent node.

3.1.3 Node Selection

For static graphs $G = (V, E)$, we define the k-neighborhood $N_v(k)$ of a node v as the set of nodes within a minimum distance of k hops from v. In the context of DTN graphs, where edges fluctuate over time, an edge $e^{i,j}$ is considered to be in E if there exists a communication link at any point between i and j with a capacity greater than zero. The parameter k is also known as the hop limit; however, it does not fully encapsulate the cost associated with transferring a segment from a Node Handler to its k-neighborhood. It is advantageous for a node in $Nv(2)$, for instance, to have a lower migration cost than another node in $Nv(1)$, especially in DTNs with highly dynamic and irregular connections.

The node selection algorithm we pursue prioritizes which nodes are optimal candidate destinations based on a comprehensive migration cost metric. The first metric, Buffer Cost, measures the amount of available buffer space reserved for incoming segments. Each node determines how much of its buffer capacity to allocate for handling segments from other nodes. The second parameter, Transmission Cost, calculates the total cost of transferring segments from the current Node Handler to one or more alternative Node Handlers. This cost depends on latency, bandwidth, and transmission schedules along the path from the current Node Handler to the destination node. The migration cost $C_{c,v}$ (l) from Node Handler node c to a k-neighborhood node v of the selected segment with length l is the weighted sum of the normalized buffer cost $Sv(l)$ and normalized transmission cost $T_{c,v}(l)$.

3.1.4. Segment Retrieval

The segment recovery algorithm determines which segment is selected for relocation to a Node Handler using a control pull operation initiated by the Node Handler. In simpler network topologies, this could involve a Node Handler retrieving a segment that it had temporarily stored at a nearby alternative Node Handler [19]. A Node Handler might issue a control request when transitioning from high congestion to low congestion. This process resembles a form of routing advertisement but, as discussed earlier, does not affect the DTN routing state. The control request for a segment could be directed to a single Node Handler or the Node Handler might utilize a form of multicast query to locate other nearby Node Handlers with segments to transfer. The decision of which segments to move now reflects the push strategy described earlier, except in this case, the decision is made by the recipient rather than the sender.

The purpose of this section is to introduce the coding scheme we aim to integrate within the DTN architecture. The fundamental principle of the coding scheme [9] is to generate a repair segment after every k source segments, where k is an integer determined by the average segment loss rate. The coding rate is then given by k/(k+1). It also includes a variable-size encoding window and may encompass segments not explicitly identified. They can be calculated using the formula:

$$R(i,j) = \sum^{j}(i,j)Bum \qquad \ldots\ldots\ldots\ldots(1)$$

where Bi...j are the segments belonging to the encoding window, and α(i,j) u is the coefficient randomly selected from the finite field Fq, used to encode the u-th segment in the repair segment R(i..j). Each repair segment simply carries the unique value of the random coefficient generator specific to itself. The source segments are transmitted unchanged (i.e., the code is efficient: implying that data information is also embedded in the encoded output), which leads to reduced coding complexity. The receiver occasionally identifies the received or decoded source segments. The retransmission rate can be scheduled either based on time or the number of received segments. The purpose of this retransmission scheme is to reduce the number of source segments involved in the encoding/decoding processes. In essence, this design enables reliable communication even if some source data segments, repair segments, or acknowledgments are lost. More importantly, in the context of DTN, the decoding does not rely on the received data segments, thus the loss recovery delay is entirely independent from the Round Trip Time (RTT). Since the decoding process is performed collectively, we do not need to check whether a given segment is lost before forwarding the other segments to the application. The issue of retransmission with ARQ is also avoided since this on-the-fly coding scheme does not require requesting retransmission. A DTN communication may involve several nodes, and as a result, segments may traverse multiple hops. In this study, we consider the single-hop case and for a comprehensive evaluation, we also examine a two-hop scenario as an example of a multi-hop case. The two-hop scenario remains sufficiently generic to be extended for multiple hops and specifically to consider rerouting situations. We elaborate on these cases in the following sections. When no Line of Sight (LoS) path exists, an intermediate node acts as a relay between the Source and the Destination [20].

Thus, this section proposes a versatile mechanism with low complexity to reduce overhead while ensuring reliable communication. Specifically, the algorithm for adaptable framework is designed based on the following considerations: The calculation operates on a hop-by-hop basis between a sender and a receiver, making it scalable to a large network. A sender-receiver pair can act as the source transmitter and relay destination, respectively. A relay node handles both roles. The

Destination accurately tracks the number of missing segments at any given time (denoted as m(t)). To ensure end-to-end reliable communication, this data is solely updated by the Destination. A relay node simply forwards this data contained in the ACK packet downstream. Link failure rates are periodically monitored at the controller. Subsequently, the receiver sends feedback indicating the loss rate and the count of missing segments back to its source. The link failure rate is computed on a hop-by-hop basis, which is simpler than end-to-end assessment. Hence, this approach is adaptable [18]. Moreover, the acknowledgment segment can be integrated with the confirmation segment to maintain the system's simplicity and reliability in the face of input channel failures.

Algorithm 1 :
Adaptive Component :
Adaptive Mechanism at recipient node:

1. Compute link misfortunate 'p'
2. If recipient is destination at that point
3. Compute missing segments 'm' in the input segment
4. else
5. m=m+1 (got from the following bounce)
6. End if
7. Send intermittent input segment including p and m
8. //Adaptive mechanism at Sender node:
    1. Let the Input segment be p and m
    2. If R is less than P at that point
    3. Send 'm' repair segments
    4. End if
    5. Value of n= $\lfloor p + q \rfloor$

Estimation at the recipient end is illustrated in Algorithm 1. The connection failure rate 'p' is periodically assessed between sender-receiver pairs. If the recipient is the Destination, it calculates the count of missing segments 'm'. Conversely, if the recipient is a relay node, it computes 'm' from the input segment received from its downstream receiver. Subsequently, the recipient intermittently transmits an input segment indicating both 'p' and 'm'. It also notifies the sender upon receipt of the input segment from the recipient containing 'p' and 'm'. If the current redundancy ratio 'R' is less than the connection loss rate 'p', the sender generates additional 'm' repair segments to adapt to changes in network conditions. The sender adjusts its redundancy ratio to the connection loss rate 'p' plus a minimum margin ϙ, ensuring the redundancy ratio approximates an integer value of n (R = 1/n). R < p may arise due to various factors such as inaccurate loss rate estimation from previous periods, dynamic changes in network conditions, or increased loss rates on new connections due to rerouting, among others [17]. In any of these scenarios, the algorithm adapts to network dynamics by sending additional repair segments to facilitate quicker recovery and thereby reduce transmission buffer size.

3.1.4 Relay node Selection Algorithm

Table 1. Notations and Meaning

Notations	Meaning
S	Node possessing the data packet which needs to be forwarded
P_d	Data Packet
RN	Relay Node
i	Relay Node Name
CH	Cluster Head
P_p	Probe Packet
P_r	Response Packet
$REU(i)$	Relative Energy Usage of node i
T^1	Transmission power constraint
RV_i	Reward value of relay node i
T_w	Waiting Time
δ	Probing cost
Q	Relay node

1. When a node, S, has a data packet, Pd, which needs to be forwarded, then it first assigns priority to the corresponding data packet. If the data packet is originated by the node S itself, then this Pd is assigned priority 1.
2. If the data packet is received from other nodes, then this Pd is assigned priority 2.
3. To determine the next RN to which the Pd can be forwarded, the CH initiates probing.
4. The CH monitors all the nodes in the neighborhood of S which can act as relay nodes in forwarding the P_d.
5. All the nodes follow a sleep-wake cycle, and hence the CH waits until each RN wakes up
6. When a relay node, i, wakes up, the CH sends a probe packet, Pp, to that RN.
7. On receiving P_p, the the node i initially estimates the REU value according to equation (1).
8. Then based on the estimated REU and Gl, the node i computes the reward value, RVi, according to equation (4).
9. After the estimation of the RVi, the relay node i checks the type of packet pending in its queue.
10. If the packet in the queue of i is of priority 1, then i will respond by sending the RVi along with the T_w to the CH.
11. On receiving the PR from i, the CH checks if the response includes T_w.
12. If the PR contains only the RVi then the CH retrieves this value.
13. If the PR includes T_w then the CH needs to wait until the relay node transmits all its priority 1 packets..
14. So, if the T_w is high then this RN, i, is not a

suitable relay node for data forwarding.

15. So, the CH waits until any other RN wakes up and then probes i to check if it can be considered for forwarding the data packet.
16. The CH collects the PR from all the probed RN.
17. Next, the CH estimates the probing cost, δ, which is the power required to send the probe packet and probe response.
18. Finally, the CH probes the surrounding RN and then selects an RN, q, with the best RVn, lesser T_w, and δ.
19. Then S forwards the probe packet Pp to the selected RN, Q.

Thus, the data packet is forwarded to a relay node selected through a power and delay-aware relay node selection process within the cluster. The selected relay node exhibits the best reward value, minimal waiting time, and also incurs minimum probing cost. This ensures efficient forwarding of the data packet within the cluster. Consequently, the relay node is chosen along the transmission path until the data packet reaches its destination efficiently.

3.1.5 Experimental Set up and Result Analysis

Table 2. Simulation Parameters

Installationenvironment	RedHat Linux
Simulator	NS2
Thenumberofnodes	100
Buffersize	50-500
Numberofgeneratedpackets	5–500
Thesizeofeachpacket	0.1(KB)
Simulationarea	10,000m×8000m
Radiorange	100m
Packetpayload	1-30KB

All simulations were conducted on the test system. Table 1 summarizes the parameters utilized in these simulations.

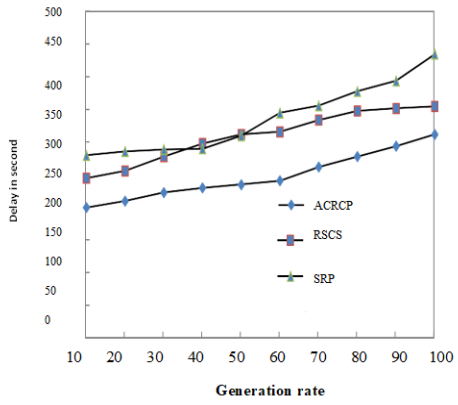

Figure 3. Generation rate Vs Delay

Figure 3. represents the output for the various existing protocols in Delay Tolerant Networks (DTNs) including Square Routing Protocol (SRP) and Remote Spooling (RSCS) Protocols. It was observed that the ACRCP protocol exhibits lower delay compared to other protocols.

4. Conclusion

In this paper, we have proposed a Power and Delay Relay Node Selection Algorithm for Cluster-based Mobile Wireless Sensor Networks. Within the cluster, when a node needs to forward a data packet to its destination, it first prioritizes the action. The cluster head then monitors all potential relay nodes. When a relay node awakens from its sleep cycle, the cluster head initiates the probing process. By analyzing the response from the probed relay node, the cluster head evaluates parameters such as reward value, waiting time, and probing cost associated with that relay node. Subsequently, the cluster head assesses these metrics for all surrounding relay nodes. The relay node that offers the best reward value, shortest waiting time, and lowest probing cost is selected by the cluster head. The node then forwards its data packet through this chosen relay node towards the destination.

In this study, we advocate for the adoption of a flexible framework to facilitate reliable communication within DTN architecture. We introduce foundational frameworks for coding and validation at relay nodes to mitigate overhead embedded in the network. Additionally, we propose a flexible mechanism with minimal complexity to reduce overhead and manage network dynamics while ensuring dependable communication. Our findings demonstrate that the proposed approach effectively reduces overhead and adapts well to changes in network conditions, including re-routing scenarios. Preliminary results suggest that this design could potentially enhance data delivery in challenging environments.

References

1. E. Altman and F. De Pellegrini, “Forward correction and fountain code sin delay tolerant networks,” in *INFOCOM2009, IEEE*, April 2009, pp.1899–1907.
2. Udayakumar, R., Khanaa, V., Saravanan, T., Synthesis and structural characterization of thin films of prepared by spray pyrolysis technique, Indian Journal of Science and Technology, V-6, I-SUPPL.6, pp.4754-4757, 2013
3. Qin, M., Zimmermann, R.: Improving mobile ad-hoc streaming performance through adaptive layer selection with scalable video coding. In: Proceedings of the 15th International Conference on Multimedia, MULTIMEDIA 2007, New York, NY, USA, pp.717– 726, 2007
4. Suresh, K.C., Prakash, S, Priya, AE & Kathirvel, A. 2015, ‘Primary path reservation using enhanced slot assignment

in TDMA for session admission', The Scientific World Journal, Article: ID405974.
5. Tournoux, P., Lochin, E., Leguay, J., Lacan, J.: Robust streaming in delay tolerant networks. In: Proc. IEEEICC, Cape Town, South Africa, pp.1–5(2010
6. Burleigh, S.: Interplanetary Overlay Network: Design and Operation V1.13. JPL D-48259. Jet Propulsion Laboratory, California Institute of Technology, 2011
7. Udayakumar, R., Khanaa, V., Saravanan, T., Chromatic dispersion compensation in optical fiber communication system and its simulation, Indian Journal of Science and Technology, V-6, I-SUPPL.6, pp.4762-4766, 2013
8. E. Koutsogiannis, F. Tsapeli, and V. Tsaoussidis, "Bundle Layer End-to-end Retransmission Mechanism," in Baltic Congress on Future Internet Communications, pp.109–115.2011.
9. J. Zinky, A. Caro, and G. Stein, "Random binary FEC scheme for bundle protocol," Internet Engineering Task Force, Internet Draft draft-zinky-dtnrg-random-binary-fec-scheme-00, 2012.
10. Udayakumar, R., Khanaa, V., Kaliyamurthie, K.P., Performance analysis of resilient FTTH architecture with protection mechanism, Indian Journal of Science and Technology, V-6, I-SUPPL.6, pp-4737-4741, 2013
11. T. de Cola and M. Marchese, "Reliable data delivery over deep space networks: Benefits of longer erasure codes over ARQ strategies," Wireless Communications, IEEE, vol.17, no.2, pp. 57–65, 2010.
12. T. T. Thai, J. Lacan, and E. Lochin, "Joint on-the-fly network coding/ video quality adaptation for real-time delivery," Signal Processing: Image Communication, vol. 29, no.4, pp. 449 – 461, 2014.
13. Udayakumar, R., Khanaa, V., Kaliyamurthie, K.P., Optical ring architecture performance evaluation using ordinary receiver, Indian Journal of Science and Technology, V-6, I-SUPPL.6, pp.4742-4747, 2013
14. D. Kim, W. Wang, N. Sohaee, C. Ma, W. Wu, W. Lee, and D.-Z. Du. Minimum data-latency-bound k-sink placement problem in wireless sensor networks. IEEE/ACM ToN, 2011.
15. Suresh, K.C. & Sivaraman K 2014 "Improvising The Performance Of Wireless Sensor Networks By Quality Aware Stream Control Transmission Protocol" International Journal of Applied Engineering Research, ISSN 0973-4562, Special issue, Vol. 9, No.22, pp.5928-5936.
16. Udayakumar, R., Khanaa, V., Kaliyamurthie, K.P., High data rate for coherent optical wired communication using DSP, Indian Journal of Science and Technology, V-6, I-SUPPL.6, pp.4772-4776, 2013
17. Myounggyu Won,Mike George, and Radu Stoleru. Towards robustness and energy efficiency of cut detection in wireless sensor networks. Elsevier AdHoc Networks, 9(3): 249–264, 2011.
18. Aruna Balasubramanian, Ratul Mahajan, Arun Venkataramani, Brian Neil Levine, and John Zahorjan. Interactive wifi connectivity for moving vehicles. In SIGCOMM, 2008.