

On the Performance of Network Coding for Multicast Data Delivery in Large Scale Mobile Ad Hoc Networks

Emeka E. Egbogah¹, Abraham O. Fapojuwo¹, Zongpeng Li²

Department of Electrical and Computer Engineering¹ / Department of Computer Science²

The University of Calgary

Calgary, Alberta, Canada

E-mail : {emeka.egbogah, fapojuwo, zongpeng}@ucalgary.ca

Abstract — The effort dedicated to the ultimate goal of designing an architectural framework for tactical mobile ad hoc networks (MANETs) has long been hindered by the excessive control overhead of routing protocols to maintain a full view of the network topology. This paper explores the viability of adopting network coding as a means of improving reliable data delivery while lowering control overhead in large scale MANETs. Simulation results show that the proposed protocol, Network Coded STORM, achieves in excess of 95% packet delivery ratio and reduces the overall overhead by more than 50% in scalability experiments.

I. INTRODUCTION

The development of mobile ad hoc networks (MANETs) continues to be an intriguing area of research; especially in the military sector where having the ability to communicate through infrastructureless and wireless means is a necessity. The MANET designed for the military sector possesses several characteristics that make it unique from other applications for which MANETs have been employed. Tactical MANETs, which can be characterized as large scale MANETs, are typically composed of thousands of mobile nodes that are organized in groups (e.g. tank squadrons) and forced to communicate over long multi-hop distances using multicast routing protocols. However, the ability for multicast routing protocols to reliably and efficiently deliver data between nodes is made challenging due to mobility, constrained radio bandwidth, limited power supply, and the excessive amount of control overhead required for each node to maintain knowledge of the entire topology of the network. Our previous work [1] proposed the Scalable Team Oriented Reliable Multicast (STORM) routing protocol to overcome the aforementioned challenges by employing a hierarchical architecture that organized a large number of nodes into teams and created multiple robust forwarding paths between each team. Although the overall performance of STORM improved upon that of other prominent protocols from the literature such as Team Oriented Multicast (TOM) [2], there still exists a pressing need to improve the reliability of data delivery while also reducing the level of high control overhead that is used to maintain topological knowledge of the network.

In recent years, network coding has been explored as a technique for improving protocol performance in a MANET [3]. While network coding has been frequently implemented in unicast transmissions for wired networks, Ahlswede *et al.* showed that network coding can also be greatly beneficial for multicast transmissions [4]. The development of CodeCast [5], a network coding based ad hoc multicast protocol, further showed that random network coding can be of great benefit for

both multicast and MANETs. CodeCast delivers near 100% of data packets and reduces overhead by as much as 50% (compared to a conventional multicast routing protocol) when evaluated in a standard MANET configuration with 100 mobile nodes in a 1500m x 1500m network area. The impressive performance of CodeCast and other network coding based protocols [6] [7] provides the motivation for investigating how network coding can improve multicast data delivery in large scale MANETs while incurring low overhead. The main contribution of this paper is the proposal of a Network Coded STORM (NC-STORM) multicast routing protocol for large scale MANETs and evaluation of its performance. NC-STORM adopts STORM for organizing teams of nodes into a robust multicast mesh structure and intelligently designates forwarding nodes to perform network coding as a means for efficiently and reliably delivering data packets between source and destination nodes. Another contribution is the performance evaluation of NC-STORM against a flat protocol that is based on multicast and network coding (CodeCast) and a hierarchical protocol that is solely reliant on multicast routing and no network coding (TOM). This paper shows how combining the properties of network coding and multicast routing for forwarding path creation can be of great benefit for performance gains in a large scale MANET. Several authors [5]-[7] have investigated the benefits of network coding in standard MANETs but, to our knowledge, there has been no published literature that explores using network coding for data delivery in large scale MANETs. The work performed in this paper serves as a contribution towards the military sector's ultimate goal of designing an architectural framework for tactical MANETs [8].

The remainder of this paper is organized as follows. In Section II, a detailed description of NC-STORM's design is presented. Section III evaluates the performance of NC-STORM using a simulation approach and discusses the results. In Section IV, the paper offers concluding statements.

II. NETWORK CODED STORM PROTOCOL DESIGN

In traditional routing, intermediate nodes often duplicate incoming packets in order to forward them to the next hop node. Over the span of several hops, a large number of duplicate packets are injected into the network when they are transmitted over multiple paths. Although routing over multiple paths creates robustness and redundancy which can lead to higher reliability, it also produces excessive overhead and limits the overall transmission efficiency of the network.

The network overhead can be lowered and the packet transmission efficiency increased by taking advantage of network coding. The intermediate nodes that are enabled with the ability to perform network coding operations do not duplicate incoming packets to be forwarded. Instead, by leveraging the broadcast nature of the wireless medium to overhear packets from different neighbors, network coding is used at the intermediate nodes to combine and encode a number of incoming packets into a single outgoing packet that has more information in it than the original packet. Fig. 1 provides an example of the benefit that network coding offers over traditional routing when data is forwarded over several hops.

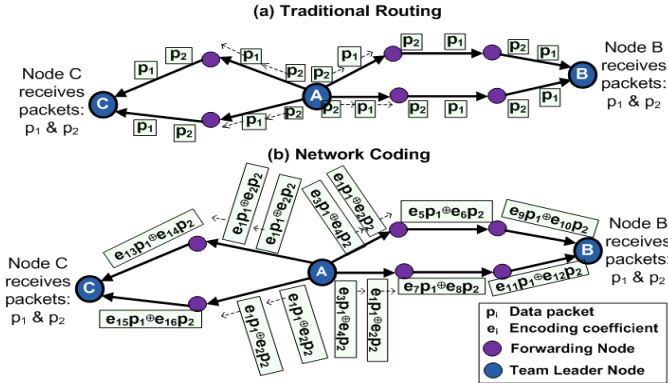


Fig. 1: Traditional Routing vs. Network Coding

In Fig. 1a, the source *node A* multicasts two data packets p_1 and p_2 over four forwarding paths to destination nodes *B* and *C*. Each packet is forwarded 6 times before it reaches the destinations, which equates to a total of 12 transmissions for p_1 and p_2 . In Fig. 1b, *node A* also multicasts the same two packets, but they are first linearly combined using network coding at the source. As a result, each intermediate node receiving the linearly combined packets only forwards a single packet that is the result of re-encoding the incoming packets with unique coefficients (e_i). Forwarding p_1 and p_2 to the destination nodes requires a total of 6 transmissions, which is twice as efficient as traditional routing. In a more sophisticated network such as a MANET where a source node must deliver packets reliably over multiple paths to destination nodes located up to 5 hops away, the efficiency advantage of network coding over packet duplication forwarding can become even greater. This efficiency advantage is clearly demonstrated in [5] where CodeCast is compared against ODMRP. CodeCast gains its advantage over ODMRP by initially setting all nodes in the network as forwarding nodes. As a result, all nodes in the network perform network coding and contribute to the end goal of forwarding packets that allows for destination nodes to reliably recover original data. Although forwarding nodes that are not regularly active in the forwarding process eventually get pruned, over the duration of a simulation, all nodes at one time or the other would have forwarded data packets. In a large scale MANET with up to 1000 mobile nodes, having every node participate in the forwarding process is an inefficient use of resources, even if network coding is adopted. The work presented in [9] suggests that not all nodes in the network are required to perform

network coding. Rather, an effective selection of intermediate nodes that can perform the network coding would be sufficient. In this section, we describe the design details for NC-STORM which is composed of two major components: (1) An algorithm to effectively select the subset of intermediate nodes located between the source and destination to function as forwarders, and (2) A network coding model to efficiently disseminate data from source to destination using the forwarding nodes set by the aforementioned algorithm.

A. Algorithm to Set Forwarding Paths

The amount of network overhead required to maintain topology information for a large number of nodes is exorbitantly high in a large scale MANET. Each node maintains a full topology view of the network in order to create routing paths between itself, the source, and the destination nodes. In STORM, the amount of overhead required to maintain routing information is drastically minimized by organizing and grouping individual nodes into teams based on their relative mobility, common interests, and ability to reach one another. Each team elects a team leader node (TLN) that is responsible for maintaining the routing information for all the TLNs in the network and also the routing information for its local team members. The routing information is attained from each TLN in the network that periodically broadcasts an update packet that details its node address, current location in the network, and to what multicast group it is subscribed to. Once all the individual nodes in the network have been grouped into teams, the TLN from each team attempts to join a multicast group using the procedures further detailed in [1]. The subscribers to the multicast group are organized into a multicast mesh structure that is composed of mutually reachable TLNs that are interconnected. An illustration of a fully connected mesh is shown in Fig. 2.

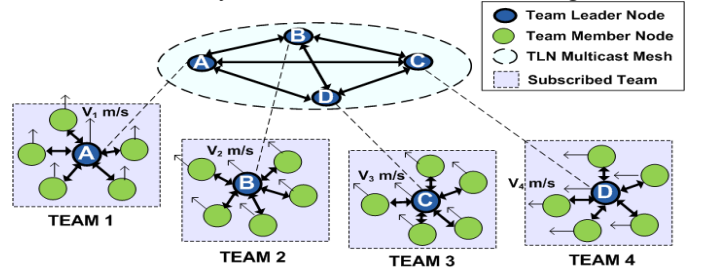


Fig. 2: Multicast Mesh Structure with 3 Connections

As also detailed in [1], TLNs exchange series of control messages (e.g. join query, join reply, connection request, etc.) between other TLNs in order to create reliable forwarding paths between each other. To ensure that the TLNs are indeed connected in the mesh in a very reliable manner, each control message (e.g. join query/reply) is transmitted hop by hop to the destination using unicast as opposed to flooding because it consumes significantly fewer memory resources, especially in a large scale MANET. If a control packet has been successfully transmitted to a next hop node using unicast, the reception of the control packet is acknowledged with an ACK packet. Once a node receives the ACK packet, it is designated as a forwarding node. Fig. 3 illustrates the forwarding node designation process.

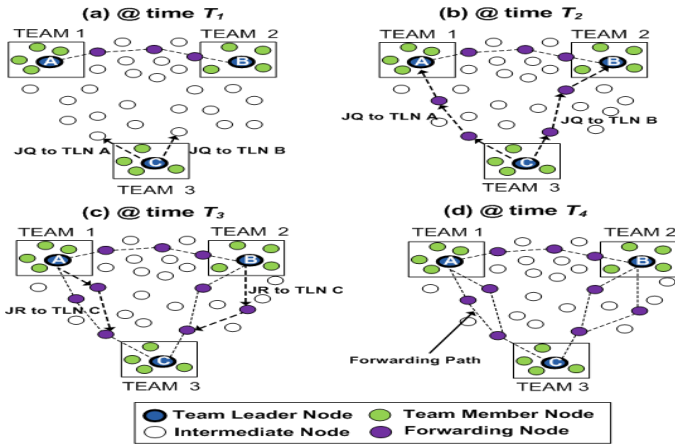


Fig. 3: Designation of Forwarding Nodes

In Fig. 3a, Team 3 wishes to join the multicast mesh that already has *TLN A* and *TLN B* connected to each other. At time T_1 , *TLN C* sends join queries to both *TLN A* and *TLN B*. In Fig. 3b, the intermediate nodes involved in the propagation of the join query packets become forwarding nodes at time T_2 . In Fig. 3c, both TLNs *A* and *B* respond to the join request with a join reply packet. Since the MANET experiences network changes between times T_2 and T_3 , the forwarding and reverse paths are different, creating more diversity for multipath creation. Finally, in Fig. 3d at time T_4 , Team 3 has successfully joined the multicast mesh and the forwarding nodes later required for data dissemination using network coding have been set. In the event that a newly designated forwarding node does not receive or propagate a control packet (e.g. join query/reply, etc.) for $t_{lmupdate}$ seconds, it is pruned as a forwarding node until it receives a new control packet. The idea for this scheme is that the nodes that are most involved in the formation and maintenance of the connections in the multicast mesh structure will be the most effective and dependable in relaying and forwarding data between team leader nodes.

B. Network Coding Model

In NC-STORM, the creation of robust and reliable forwarding paths allows for network coding to be performed at the source, intermediate, and destination nodes that partake in the data dissemination process. In the subsections below, a network coding model is defined to facilitate network coding operations in the NC-STORM nodes.

1. Sending a Network Coded Packet from a Source Node

At a source node, each set of N sequentially generated application messages are buffered into memory. Once the N^{th} application message has been buffered, the source node randomly generates N encoding coefficients over the Galois Field $GF2^N$ in order to linearly combine the N buffered application messages. This process is known as random network coding. The most suitable value for N is determined by the node density characteristics of the network. Reference [10] has shown that $N=8$ provides the best coding scheme when the number of neighbors is greater than 11. In our case, each node from the network we use for simulations has an average of 13 neighbors. Therefore, we select $N=8$ for our

coding scheme. The linearly combined application messages, also called coded packets, are grouped into a block called a ‘generation’. Each generation block is identified by $block_{GenerationID(GID)}$ to denote which application message belongs to which block. An application message m_k is assigned to $block_{GID}$ if its sequence number k is greater than or equal to GID but less than $GID+N$. The first generation block created has a GID of 1, and each subsequent block is incremented by N . Each coded packet that is transmitted to the network layer is appended with a header that holds GID , N , $rank$, and the encoding vector. The GID and N combine to specify which generation block the coded packet belongs to. The $rank$ is the number of generation blocks that are combined to yield a coded packet. The encoding vector holds all the randomly attained N encoding coefficients. The main purpose of transmitting the encoding vector with the coded packet is to ensure that when it is received at an intermediate node, it is linearly independent with the coding vectors locally held at the intermediate node. Linear independence is a function of uniformly selecting encoding coefficients randomly over a Galois Field in a completely independent and decentralized manner. With the use of random network coding, there is a certain probability of selecting linearly dependent combinations. However, results from the literature indicate that even for a small field size (2^8), which is used in this paper, the probability becomes negligible [9]. Once a full $block_{GID}$ has been assembled, N number of coded packets c_{GID} are generated and broadcast to the neighboring nodes.

2. Handling a Network Coded Packet at a Forwarding Node

When a forwarding node receives a coded packet with a new GID , it stores the packet in local memory and starts a timer that expires in $block_{timeout}$ seconds. In the time before the timer expires, the forwarding node collects coded packets belonging to $block_{GID}$ that have encoding vectors that are linearly independent of each other. Once the timer expires, the forwarding node uses the stored coded packets to build a newly re-encoded packet. Re-encoding is simply performed by again randomly generating N encoding coefficients and linearly combining them with the stored coded packets. If the forwarding node only has less than N number of received coded packets, it records this number as the $rank$ in the new header, and uses the packets it does have to perform re-encoding. After local re-encoding has completed, a single recoded packet is broadcast to the neighbors. Each coded packet belonging to $block_{GID}$ that originates from the source node has a network lifetime of $packet_{deadline}$ seconds. If this lifetime has been exceeded when processed at any forwarding node, the coded packet is discarded.

3. Handling a Network Coded Packet at a Destination Node

Unlike forwarding nodes, destination nodes decode incoming packets. Destination nodes are nodes that belong to a team that has subscribed to a multicast group. When a destination node receives a coded packet with a new GID , it stores the packet in local memory and starts a timer that expires in $block_{timeout}$ seconds. To decode and recover the original N application messages belonging to $block_{GID}$, a node

must collect at least N number of coded packets of the same GID , and have encoding vectors that are linearly independent of each other. If the encoding vectors are dependent, the decoding process may result in the loss of application messages. Linear independence assures that the encoding values used to combine the packets result in a non-zero determinant when the vectors are being decoded. Once the necessary number of coded packets has been received, they can be simply decoded using Gaussian elimination to extract the original application messages. However, if the timer expires and the destination node has not received N number of coded packets, it forwards the re-encoded packets to its neighbors using the same methodology as in the previous section. If the destination node is the TLN and it receives N number of coded packets and successfully decodes and recovers the original number of application messages, it stores the application messages in memory and uses them for future localized packet recovery, if needed.

4. Localized Packet Recovery

If a TLN successfully decodes and recovers the original application messages sent by the source node, it delays a certain time period before broadcasting a message to its team members indicating it has successfully received a set of application messages. The destination nodes that have not been able to recover the particular set of application messages reply with a message that is delivered via unicast to alert the TLN that it is in need of packet recovery. If 80% or more of the team members request packet recovery, the particular application message is broadcast as a single data packet. Otherwise, the requested data packet is unicast to each requesting team member. If a TLN receives new coded packets belonging to a new $block_{GID}$ without successfully decoding the previously received coded packets belonging to $block_{GID-N}$, it is an indication that the TLN requires additional ‘innovative’ packets to help decode the application messages of $block_{GID-N}$. Innovative packets are coded packets that carry information that contributes to the encoding/decoding process. Non-innovative packets are typically linearly dependent and thus discarded. To help recover the application messages, the TLN broadcasts a single message requesting that its first hop neighbor send a single coded packet to it.

III. PERFORMANCE EVALUATION

A. Simulation Setup

The performance of the proposed NC-STORM protocol is evaluated using QualNet 3.9.5. In our simulations, each source generates data in a constant bit rate (CBR) fashion with the user datagram protocol (UDP). We use the IEEE 802.11 MAC with distributed coordinated function (DCF) and the two-ray ground path-loss model with no additional fading for the physical layer. The transmission range of each node is 376m and the channel bandwidth is 2Mbps. In addition to NC-STORM, we implemented two other protocols for MANETs: CodeCast and TOM, and validated our implementations against those in the published literature. CodeCast represents a purely network coding multicast protocol while TOM only uses multicast routing and no network coding. For the

configuration of the network, 1000 nodes are uniformly placed within a 6000m x 6000m terrain. The network is divided into 36 teams (average of 25 members per team) where each team moves following the reference point group mobility (RPGM) model [11]. For maintaining a view of the network topology, NC-STORM and TOM use 1 second intervals for TLN/TRN table updates. For the network coding simulation model, both NC-STORM and CodeCast use 80 milliseconds for *blocktimeout* and 2 seconds for *packetdeadline*.

Each simulation was executed 10 times using random seed numbers and the results were averaged over those runs. The results are presented with 95% confidence intervals, only shown for NC-STORM. The simulations for NC-STORM and TOM were executed for 100 seconds while CodeCast ran for 120 seconds with randomly chosen multicast source(s) and destination teams. In each experiment, a single source transmits packets at a rate of 10 packets per second with each packet having a size of 512 Bytes. The performance is measured by the following metrics: (1) Packet Delivery Ratio (PDR) - Ratio of Number of Data Packets Sent to Number of Data Packets Received, (2) Normalized Packet Overhead - Number of Data Packets Transmitted per Data Packet Delivered, and (3) End-to-End Delay - Average time to transmit a packet from the source and have it decoded at the destination.

B. Experiment #1: Effect of Mobility on Performance

In this experiment, the node speed is varied across {0, 10, 20, 30} m/s and seven randomly selected teams are subscribed to a single multicast group. Figs 4 and 5 show the simulation results for the protocols as the node speed is increased from 0 m/s to 30 m/s for PDR and delay, respectively. In Fig. 4, NC-STORM consistently outperforms both CodeCast and TOM when the node speed is 30 m/s or less. The reason NC-STORM shows a highly favorable performance is that the nodes selected as forwarding nodes form highly reliable data forwarding paths between the source and destination nodes despite increased node speed. These reliable data paths are formed when multicast mesh connections are created between TLNs. As the speed of nodes increase and old forwarding paths become partitioned, new intermediate nodes are designated as forwarding nodes to ensure that data continues to be delivered reliably. However, when the node speed exceeds 20 m/s the PDR of NC-STORM drops by more than 10% because the forwarding paths between the source and destination nodes do not update fast enough to deal with the changing team speeds. In the case of CodeCast, an increased number of forwarding paths are created as the network conditions change due to increased node speed. As a result, nodes that were previously pruned for lack of activity become forwarding nodes and help CodeCast achieve a PDR that improves with increased node speed. In Fig 5, packets transmitted using both NC-STORM and CodeCast experience an average end-to-end delay that is approximately 10 times higher than TOM. In our implementation of network coding, *blocksize* number of application messages belonging to the same generation is buffered for an average of 0.35 seconds before a coded packet can be sent to the network layer.

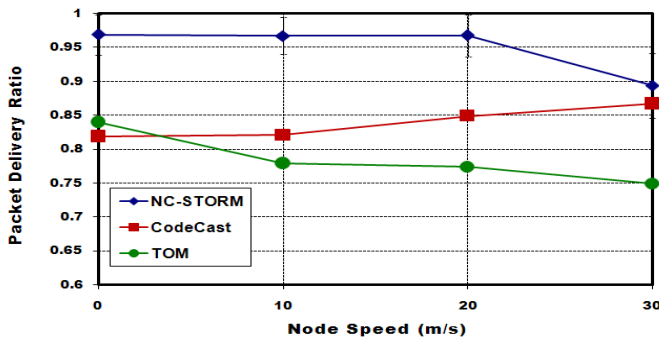


Fig. 4: Packet Delivery Ratio vs. Node Speed

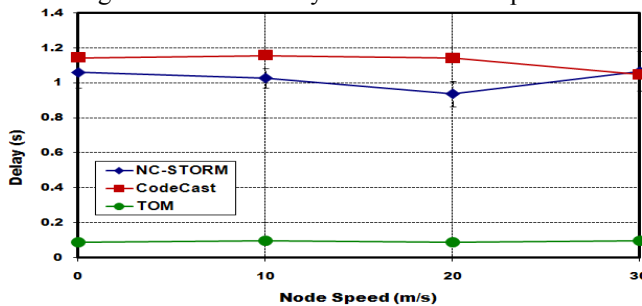


Fig. 5: Delay vs. Node Speed

In addition, each neighboring node receiving the broadcasted coded packet adds to the end-to-end delay by storing, re-encoding, and decoding the packets. This process consumes approximately 80 milliseconds at each node on the forwarding path.

C. Experiment #2: Impact of Scalability on Performance

To test scalability, the number of teams is varied across the set $\{5, 7, 9, 11\}$ which corresponds to $\{140, 196, 252, 308\}$ destination nodes. The node speed is set constant at 20 m/s. Figs 6 and 7 shows the simulation results for the protocols' PDR and normalized packet overhead, respectively, as the number of teams is increased. Fig. 5 shows that the PDR for NC-STORM stays consistently high ($> 95\%$) as the number of teams is increased to 11. As the number of teams increases, the underlying multicast mesh becomes better connected and therefore more forwarding paths are made available for data to be sent to destination nodes. Since NC-STORM expends a low amount of overhead in transmitting data between a large number of multicast members, as is shown in Fig. 7, the network is less prone to congestion and thus a high percentage of data packets can be delivered. The PDRs for CodeCast and TOM are approximately 10% and 20% lower than that for NC-STORM, respectively. The lower PDR can be attributed to the fact that CodeCast and TOM encounter more congestion and thus dropped packets, due to the high proportion of nodes that are responsible for forwarding.

IV. CONCLUSIONS

This paper proposes the NC-STORM protocol which incorporates network coding into STORM. The results show that network coding is viable in large scale MANETs because NC-STORM delivers a high percentage of data packets while

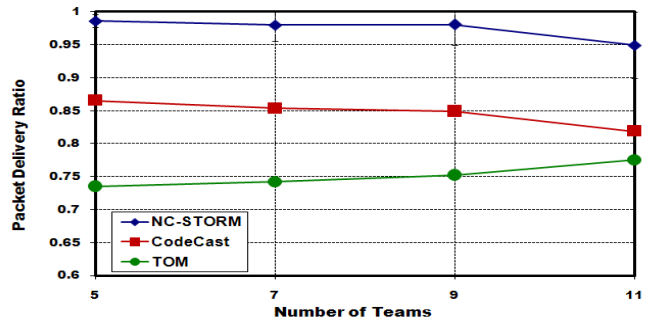


Fig. 6: Packet Delivery Ratio vs. Number of Teams

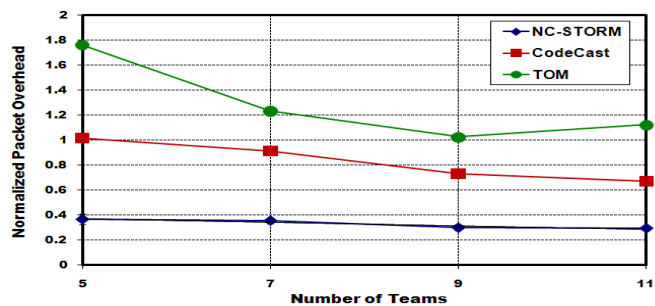


Fig. 7: Normalized Packet Overhead vs. Number of Teams

incurring low overhead in scenarios where mobility speed and the number of multicast members are steadily increased. Therefore, network coding for data dissemination can be further investigated as part of designing an architectural framework for tactical MANETs.

REFERENCES

- [1] E. Egbogah, A. Fapojuwo, and N. Chan, "Scalable Team Oriented Reliable Multicast routing protocol for Tactical Mobile Ad hoc Networks", *Proc. of IEEE MILCOM '08*, pp. 1-7, San Diego, Nov. 2008.
- [2] Y. Yi, M. Gerla, JS. Park, and D. Maggiorini, "Team-oriented Multicast: A Scalable Routing Protocol for Large Mobile Networks", *Proc. Of NGC '03*, pp. 143-154, Munich, Sep. 2003.
- [3] T. Kagi and O. Takahashi, "Efficient Reliable Data Transmission Using Network Coding in MANET Multipath Routing Environment", *Proc. Of KES'08*, pp.183-192, Zagreb, Sep. 2008.
- [4] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network Information Flow", *IEEE Transactions on Information Theory*, IVol.4 Issue 46, pp.1204-1216, Jul. 2000.
- [5] J.S. Park, M. Gerla, D. Lun, Y. Yi, and M. Medard, "CodeCast: A Network Coding based Ad hoc Multicast Protocol", *IEEE Wireless Communications*, Vol.13, pp.76-81, Oct. 2006.
- [6] S.Y. Oh and M. Gerla, "Robust MANET Routing using Adaptive Path Redundancy and Coding", *COMSNET'09*, pp. 1-10, Bangalore, Jan. 2009.
- [7] D. Koutsonikolas, Y. C. Hu, C.-C. Wang, "Pacifier: High-Throughput, Reliable Multicast without "Crying Babies" in Wireless Mesh Networks", *Proc. Of INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009.
- [8] A. Helmy, "Architectural Framework for Large-Scale Multicast in Mobile Ad Hoc Networks", *IEEE ICC '02*, Vol.4, pp. 2036-2042, New York, Apr. 2002.
- [9] D.S. Lun, M. Medard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks", *Physical Communication*, Vol.1, pp. 3-20, Mar. 2008.
- [10] I-H. Hou, Y-E. Tsai, T.F. Abdelzaher, and I. Gupta, "AdapCode: Adaptive Network Coding for Code Updates in Wireless Sensor Networks", *INFOCOM*, Phoenix, Apr. 2008.
- [11] X. Hong, and M. Gerla, "Dynamic Group Discovery and Routing in Ad Hoc Networks", in *Proceedings of the First Annual Mediterranean Ad Hoc Networking Workshop*, Sardegna, Sep. 2002.