Towards Bandwidth Efficient TDMA Frame Structure for Voice Traffic in MANETs

Naresh Vattikuti, Mallesham Dasari, Himanshu Sindhwal
UURMI Systems Pvt. Ltd.
Hyderabad, India.
Email: {vvsnaresh, malleshamd, himanshus}@uurmi.com

Bheemarjuna Reddy Tamma
Department of CSE, IIT
Hyderabad, India.
Email: tbr@iith.ac.in

***Abstract*—Time division multiple access (TDMA) based channel access is one of the most widely used mechanisms in Mobile Ad hoc Networks (MANET) with its efficient channel access and collision free transmissions at its core. In this paper, we propose a unique TDMA frame format designed specifically for supporting two-way voice communication in bandwidth efficient manner. The proposed algorithm uses single radio channel for both control and data message exchanges and achieves multi-hop communication. As the transmission of control messages in the same channel leads to overhead, we propose a mechanism to share the control slots between the nodes for reducing the control overhead. The TDMA frame design shows the optimized channel sharing based on the periodicity and payload size of a typical voice packet. The experimental analysis is performed on Network Simulator (NS-3) as well as on Linux kernel with Ethernet emulation of wireless network. The experimental results clearly show considerable improvement in bandwidth efficiency over the existing protocols.**

***Keywords*—TDMA, Control Overhead, Voice call, QoS, MANETs.**

I. Introduction

TDMA slot assignment in MANETs is an efficient MAC protocol due to its power efficiency and deterministic nature. Many wireless systems such as GSM and LTE have centralized slot assignment, where a base station allots slots to each node for data transmission. Due to its limitations such as single point of failure and congestion at the base station, the distributed implementation has become a challenging research area in MANETs. Here, the nodes in the network exchange the node information with their neighbors and perform the slot assignment in a self configurable manner. For this, many protocols such as [1] use different radio channels for control and data communication. Using a single carrier for both control as well as data communication is a challenging task where the bandwidth should be effectively used with minimum control overhead. In this work, we have concentrated on reducing the control overhead and designing a frame structure for effeciently supporting voice calls in a TDMA based distributed multi-hop MANET using a single channel.

Voice is the most used application in MANET scenarios such as military, vehicular, disaster management, etc. The scope of this paper is to present a TDMA Frame Structure (TFS) designed specifically for effeciently supporting voice calls and minimizing the control overhead in a distributed MANET with each node having a single radio channel for communication. We have implemented the TDMA with our unique TFS on NS-3 simulator. Also, one of the highlights of our work is TDMA implementation by Ethernet emulation at Linux kernel level. Hence, in this work, we talk about the proposed TFS and control slot sharing between the nodes for bandwidth optimization. Further, we present an overview of how the Linux Ethernet emulation is created with four nodes connected to a hub. Rest of the paper is organized as follows: Section II discusses the related work done in existing TDMA literature. Section III describes the proposed work. The experimental setup and performance evaluation are described in Sections IV and V respectively. The conclusions are presented in Section VI.

II. Related Work

In a given TFS, lesser the control information exchange, more the bandwidth efficiency. The authors of [1, 2] perform the slot assignment using control phase with bootstrap slots which are dedicated to a node on frame basis, which at times consumes more bandwidth for control information. Dynamic TDMA slot assignment [3], in order to minimize the control overhead, does not feature any control phase per frame, instead when a node comes, it will request its neighbors to switch to control phase and switch back to data phase after node joins, which results in delay in node joining and disturbance in data communication during this time. This is also not very suitable for a typical multimedia communication which imposes strict QoS constraints. The authors of [4] proposed voice communication based TDMA protocol with minimum number of nodes and minimal topology changes for small group communications but it is a centralized approach. The authors of [5] proposed a bandwidth reservation protocol by on-demand Quality of Service (QoS) routing. This, however is not minimizing the control packet overhead for effective utilization of channel access which will support more number of simultaneous voice calls. In [6, 7], the authors proposed algorithms for increasing the capacity of the system by enhancing the data slots usage. However, the work does not address how much control transmission overhead is controlled.

In this paper, we propose an efficient implementation of dynamic sharing of the control slots for minimizing the control overhead and a TDMA frame design that includes slot duration, frame duration, specifically designed for packet switched voice calls.

III. Proposed Work

The importance of effective utilization of bandwidth with reduced control overhead and TFS is explained in Sections I and II. Here, we first discuss the Control Slot Sharing (CSS)

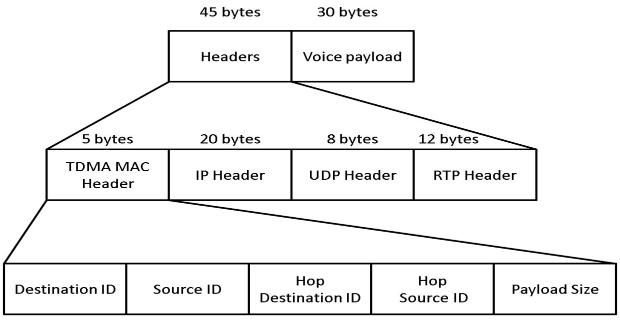

Fig. 1: **TDMA Packet Structure**

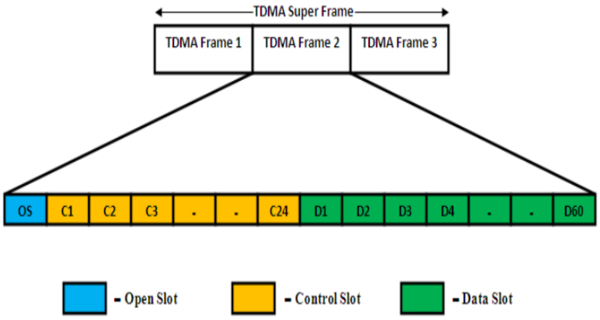

Fig. 2: **Proposed TFS**

for bandwidth efficiency and then the unique TFS design for voice traffic.

A. CSS

Slicing a channel into time slots and assigning it to the required nodes, conventionally gives shape to a TDMA frame. Typically, the time channel is divided into frames which is in turn divided as slots. Also, the TDMA frames are grouped together and called as a super frame. The first design consideration in our approach is reducing the control overhead. This, we achieved by making a node use a control slot once in a super frame instead of every frame. Hence, two or more different nodes can have same control slot number but they will use it in different frames of the super frame without any contention. The number of control slots in a super frame is equal to the maximum number of nodes that the network can accommodate in a 2-hop region. When a node wants to joins the network, first it will listen and based on the control packets received it will choose a control slot number (C) and control slot frame number (F). This means the node will send its control packet in the slot 'C' of frame 'F' of each super frame.

B. TFS

The slot duration is another crucial parameter in the design. Here, we present a unique TFS designed specifically for voice traffic and derive the frame and slot durations based on the voice traffic's periodicity which can vary in multiples of 10ms. The slot duration can be calculated effectively for a voice packet because most of the voice coders produce the voice packets with constant bit rate. According to the above mentioned TFS, TDMA frame and slot durations can be configured in such a way that the durations are good enough for two-way voice traffic. Following values assume channel bandwidth as 1 Mbps. Considering G.729 codec 6 Kbps data rate and periodicity of the packets as 40ms. The payload of the voice packet is calculated from the Eqn (1).

$$\begin{aligned} Voice_{payload} &= T_{br} * F_d \\ &= 6 * 1024 * 40/1000 \\ &\approx 30Bytes. \end{aligned} \tag{1}$$

where, T_{br} is the bit rate of the codec and F_d is the packet interval of the codec. The total size of voice packet is calculated using Eqn (2) and Eqn (3).

$$Packet = Voice_{payload} + Hdrs_{size} \tag{2}$$

$$Hdrs_{size} = \sum(RTP_{hdr}, UDP_{hdr}, IP_{hdr}, MAC_{hdr}) \tag{3}$$

Hence, Packet= 30+12+8+20+5 ($Voice_{payload}$ + RTP_{hdr} + UDP_{hdr} + IP_{hdr} + MAC_{hdr}) = 75 bytes per 40ms as shown in Fig-1. Our TDMA MAC header consists of 5 bytes:'Destination ID' which is the node id of final destination of packet, 'Source ID' is the node id of the node which generated the packet, 'Hop destination ID' is the node id of the next hop, 'Hop source ID' is the node which currently has the packet and last byte contains the payload length. From above information, the TFS for a typical voice packet is designed in an efficient way by choosing appropriate slot durations for control and data slots in the TDMA frame. Table I describes the slot durations of the proposed TFS. From Table I and Eqns (1) to (3), data slot duration is 625μs (including 16μs of guard period) and a maximum of 78 bytes can be transmitted in 609μs. The Control slot duration is 100μs which is derived based on the control packet size to be sent in the control slot. As shown in Fig-2, one superframe consists of 3 TDMA frames of 40ms each. Each TDMA frame consists of 1 open slot which is used by a new node in joining the network, 24 control slots for control transmissions and 40 data slots. The described frame structure gives a control overhead of mere 6.25%. Also, since a node will use a control slot once in a super frame, 72 nodes (24*3 (number of frames in a super frame)) can be supported in a 2-hop neighborhood. Since control slots will be spatially reused, there is no limit on the

TABLE I: **Parameters based on voice packet periodicity**

Slot information	Quantity
Number of Open slots	1
Number of Control slots	24
Number of Data slots	60
Open slot	100 μs
Control slot	100 μs
Data slot	625 μs
Guard time	8+8 μs
Frame duration	40 ms
Super Frame duration	120 ms

total number of nodes in the complete network but in a given 2-hop region only 72 nodes can be supported with this frame structure. A virtual circuit is established for each flow with data slot allocation and node releases the slot in the absence of the flow attached to the slot. The requesting node knows about its 2-hop slot allocation through control messages from its neighbors and chooses a free data slot.

C. Capacity analysis in various Network topologies

The number of voice flows between a single pair of nodes in various network topologies is calculated as:

a. Single-hop Network

60 data slots supports maximum of 30 two-way simultaneous voice calls in a single-hop scenario as each two-way voice call requires two slots (one slot each for caller and callee).

b. Two-hop Network

In a two-hop scenario, each 2-hop call takes 4 distinct data slots (1 for caller and callee each and 2 slots for the intermediate node for forwarding packets in both directions). Hence, it gives a maximum of 60/4 (i.e. 15) two-way simultaneous voice calls.

c. Three hop Network

Each 3-hop call takes 5 distinct data slots as source and destination will require 1 slot each and that too they can use same slot since both are 3-hops apart while the two intermediate nodes will require 2 slots each for forwarding packets in both directions. Hence, it can support upto 60/5 (i.e. 12) two-way simultaneous voice calls as the spatial reuse of the slots comes into picture.

c. Four or more hops network

For calls of four or more hops, each call takes 6 distinct data slots. Hence, it gives a maximum of 60/6 (i.e. 10) two-way simultaneous calls.

IV. Experimental Studies

The performance of the proposed work is investigated by implementing the TDMA with 150 nodes on NS-3 simulator. Also, unlike the previous works [2-5][12], we implemented the algorithm with a cross layer approach [9] at the Linux kernel as in [10], to test the performance of the algorithm in the real-time environment. Hence, the experimental results are shown with respect to simulation environment on NS-3 and a real-time implementation at the kernel level on an emulated Ethernet overlay network environment. The performance of the proposed work is investigated by comparing the control overhead (Υ_{co}) and end-to-end delay with respect to the hop count and the number of nodes in the network. The definition of Υ_{co} is taken from the Eqn (4), where C_p is the total number of control packets sent over the super frame and D_p is the total number of data packets sent over super frame. The performance results are compared with USAP protocol[1] and variation of super frame length, i.e number of frames in a super-frame (S_α).

$$ControlOverhead(\Upsilon_{co}) = C_p/(C_p + D_p) \qquad (4)$$

A. NS-3 Simulation

The simulation setup consists of 150 nodes arranged with a 15-node neighborhood degree over 10 hops over a 100x150 units terrain area with a transmission distance of each node equal to 1 unit. The simulation parameters considered for the experiment are same as that are derived in Table I.

TABLE II: **Linux Ethernet Emulation Parameters**

Number of Linux nodes	4
Openc slot duration	2ms
Control slot duration	2ms
Data slot duration	3ms
Guard time	200 μs
Number of open slots	1
Number of control slots	4
Number of data slots	10

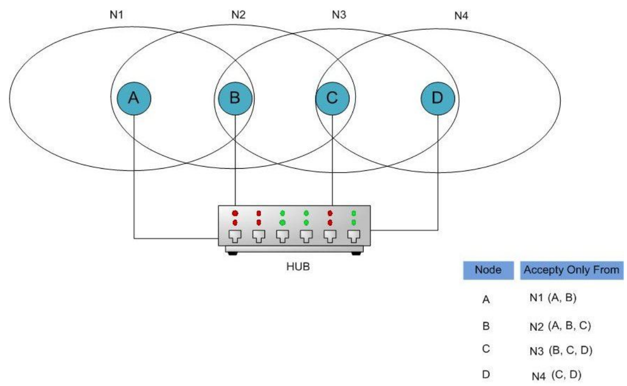

Fig. 3: **Linux Ethernet Overlay Network for Emulation**

B. Linux Ethernet Emulation

The Ethernet emulation setup is made using an overlay network with four systems having linux installed and connected using Ethernet cables to a hub. Here, on each system, we created two interfaces called $tdma0$ and $eth0$. As the $tdma0$ interface is not connected to any radio interface in this setup, we make use of Ethernet interface ($eth0$) for emulating the radio interface. Any application sends data onto $tdma0$ interface which goes upto our TDMA MAC. On $tdma0$ interface, we bypassed the CSMA/CD MAC and underlying physical layer functionality and our TDMA MAC receives packets from the network layer to $tdma0$ interface. At our TDMA MAC, we created a kernel to user space communication using Linux $Netlinksockets$ [11] which is used for bypassing the MAC layer packet to physical layer and to send the packet to a user application (UDP) waiting on the corresponding $netlink\_socket$ in the user space. This UDP application receives the data from kernel(i.e, from our TDMA MAC at the kernel space) and transmits the data through $eth0$ interface. Similarly, on the receiving side, a UDP application listens to the incoming packets on $eth0$ interface and filters the packets as shown in Fig-3, and sends the packets to kernel using $netlink\_socket$ which gives packets to our TDMA MAC.

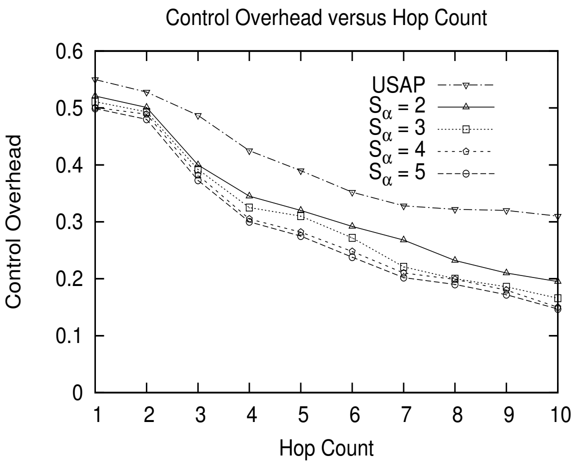

Fig. 4: **Control Overhead with respect to Hop count**

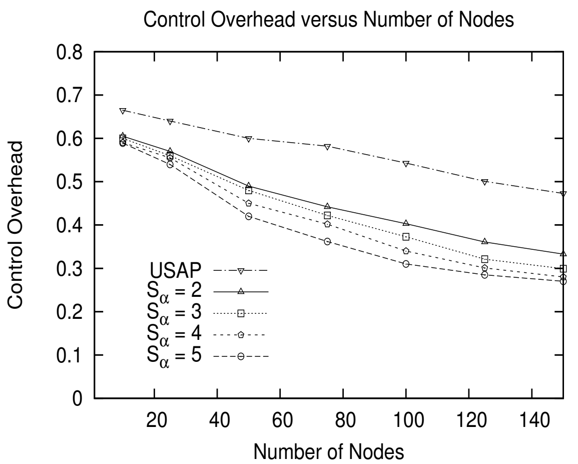

Fig. 5: **Control Overhead with respect to Number of Nodes**

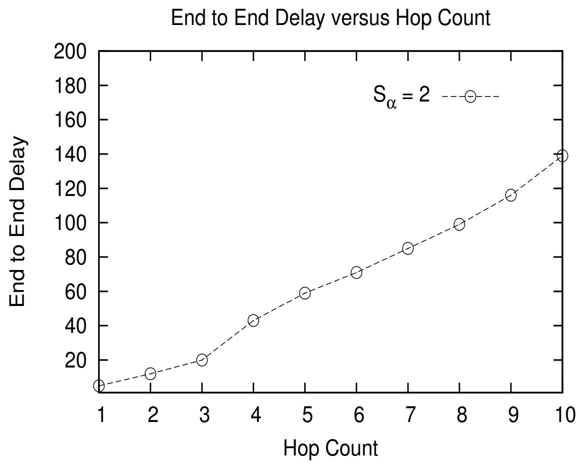

Fig. 6: **End-to-end delay with respect to Hop count**

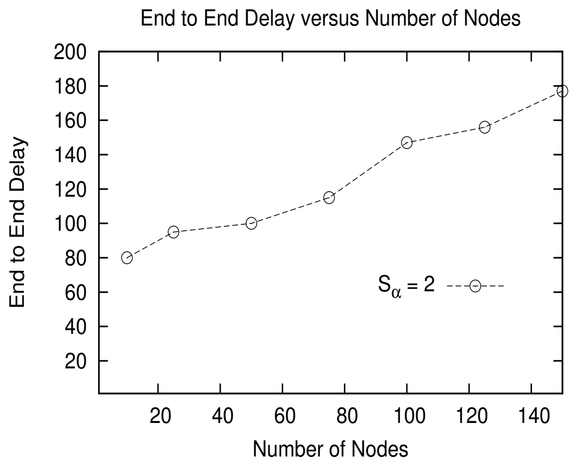

Fig. 7: **End-to-end delay with respect to Number of Nodes**

The experimental setup consists of four nodes inserted in a string topology as shown in Fig-3. Node-A has visibility of Node-B, Node-B has Node-A and Node-C, Node-C has of Node-B and Node D, and Node-D has of Node-C, which makes it a 2-hop network. Some of the configuration parameters such as slot durations are much higher as shown in Table II than the values detailed in Table I due to the hardware limitations in our experimental setup. After each node gets its dedicated control slot on a super frame basis, we make a voice call from Node-A to Node-D which will trigger following sequence of events on the nodes in the order explained below (assuming route is already available):

- As soon as Node A has some data to send it requests and gets slot number 6 (first slot in data slot pool) as its data slot and sends the data to Node-B.
- On receiving data from Node-A which needs to be forwarded to Node-C (which is the next hop), Node-B requests and gets slot number 7 as its data slot and forwards data to Node-C.
- On getting data from Node-B, Node-C in turn requests and gets slot number 8 as its data slot and data is sent to Node-D.
- Data reaches Node-D and in order to reply back it acquires a data slot same as Node-A, i.e. slot number 6 thereby spatially reusing that slot.
- Now the reply from Node-D reaches Node-C, which requests slot 9 (although it already has one slot but the flow i.e. (source, destination) ordered pair is different) and forwards data to Node-B in slot 9.
- Like Node-C, Node-B also acquires slot number 10 and forwards the data to Node-A. In total, this 3-hop 2-way call required 5 data slots (namely 6,7,8,9,10). This completes the 2-way communication between Node-A and Node-D.

As the real-time environment requires clock synchronization, the system makes use of the clock drift reduction method proposed in [8].

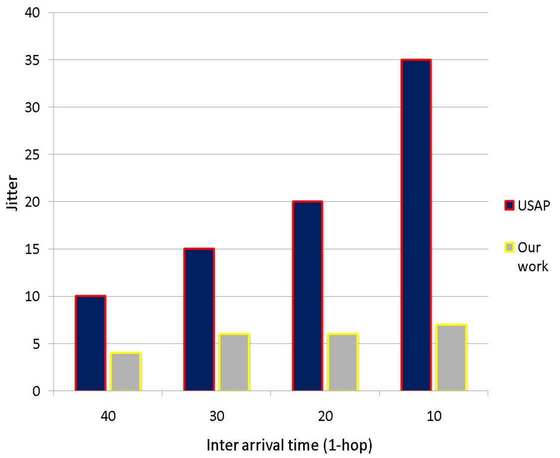

Fig. 8: **Jitter versus packet interval (single hop)**

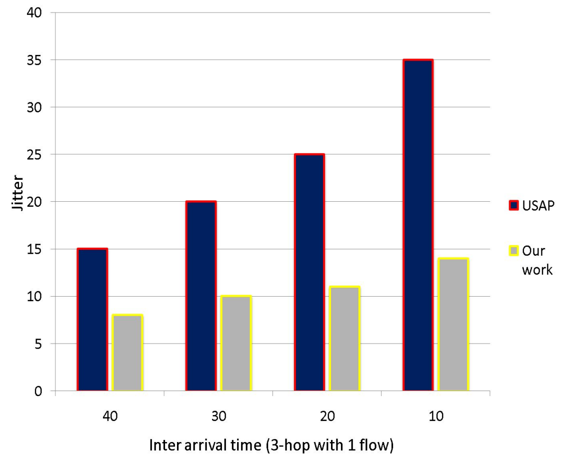

Fig. 10: **Jitter versus packet interval (3-hop with 1 flow)**

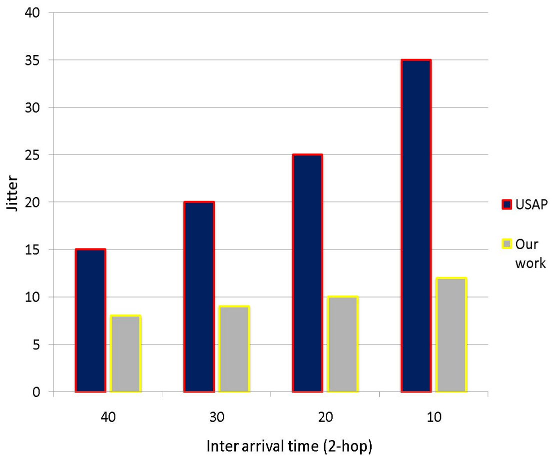

Fig. 9: **Jitter versus packet interval(2-hop)**

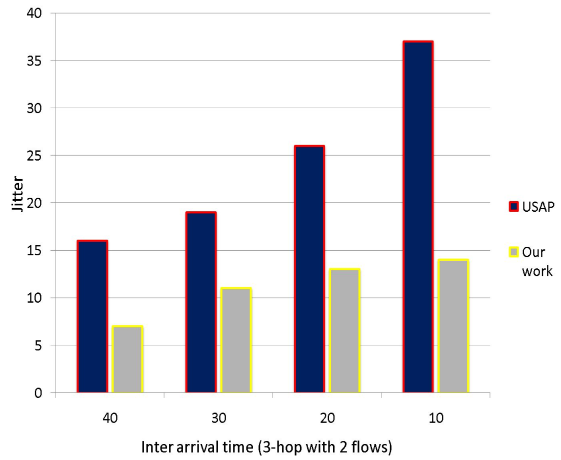

Fig. 11: **Jitter versus packet interval (3-hop with 2 flows)**

V. Experimental Results

A. NS-3 Simulation

As shown in Fig-4, Υ_{co} value with respect to hop count falls to very low value in the case of proposed approach with different S_α values when compared to USAP protocol. As the number of nodes in the network increases, the relative control overhead with respect to actual data is also reduced which can be observed in the Fig-5. Additionally, the end-to-end delay over multiple hops is also analyzed with proposed algorithm by keeping the S_α value to 2. The curves shown in Fig-6 and Fig-7 show end-to-end delay with respect to hop count and number of nodes in the network, respectively.

B. Linux Ethernet Emulation

We compared the variance in delay with respect to flows in single hop, 2-hop and 3-hops, which is crucial factor for achieving the quality of voice in a wireless network. As the TFS is specifically designed for voice applications and slots in such manner, the observed jitter values are very less as shown in Fig-8 to Fig-11(Fig-8 for single hop flows, Fig-9 for 2-hop flows, Fig-10 for 3-hop with one flow and Fig-11 for 3-hop with 2 flows). These figures show the jitter experienced for different values of inter-packet arrival time. As the frame structure is designed based on the packet interval of voice codec, the queuing delay for voice packets is very low which results into minimal variance in delay. Also as expected, jitter is minimum for inter-packet arrival time of $40ms$ as it exactly matches the frame duration. We also calculated the control overhead with respect to different hop-count as shown in the Fig-12. Here, the control overhead is lesser for 3-hop flows as compared to 1-hop and 2-hop flows, as the number of data transmissions involved will be more. Also, Fig-13 shows the end-to-end delays with voice payload which are well below the 150ms mark recommended by ITU-T G.114. For single-hop communication, if application sends the data just after the end of the slot, then it waits for one frame duration for its slot to arrive, hence the maximum delay is close to one frame duration (40 ms). Similarly, 2-hop and 3-hop results are shown for best, average and worst cases.

VI. Conclusions

In this paper, we proposed an efficient frame structure designed specifically for voice communication with reduced

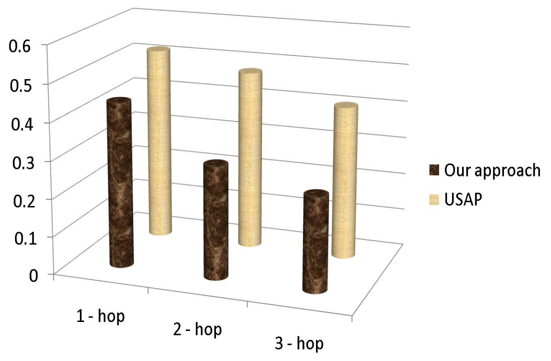

Fig. 12: **Control Overhead (Υ_{co}) versus Hop Count**

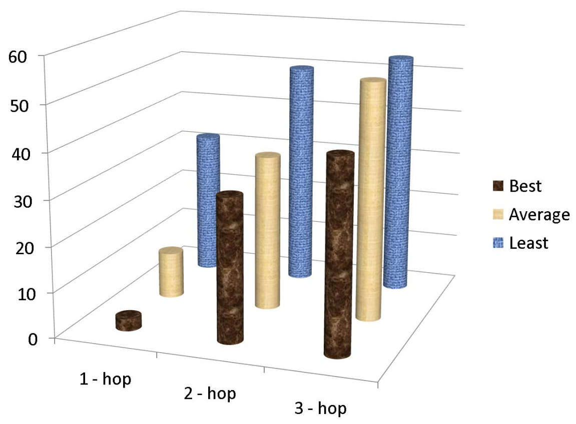

Fig. 13: **Variation of End-to-end delay versus Hop Count**

control overhead. We have shown results of reduced control overhead over USAP. Also, we have shown results with two different experimental setups for analyzing the performance. The slots distribution can be further optimized for voice traffic in silence periods of voice applications. Typically, voice applications send high frequent CNG (Comfort Noise Generation) packets when there is no voice activity. Hence, whenever CNG packets are active, there is scope for optimization in data slot assignment.

REFERENCES

[1] C. D. Young, "USAP: a unifying dynamic distributed multichannel TDMA slot assignment protocol," in Proc. IEEE MILCOM 96, vol. 1 (Oct. 1996).

[2] C. D. Young, "USAP multiple access: dynamic resource allocation for mobile multihop multi-channel wireless networking," in Proc. IEEE MILCOM 99 (Nov. 1999).

[3] Akimitsu Kanzaki, Toshiaki Uemukai, Takahiro Hara, Shojiro Nishio, "Dynamic TDMA Slot Assignment in Ad Hoc Networks", IEEE Computer Society 2003.

[4] Sang-Ho Hwang; Byoungchul Ahn, "A TDMA protocol design to relay voice communications," Communications, Computers and Signal Processing (PACRIM), 2013 IEEE Pacific Rim Conference on, 27-29 Aug. 2013.

[5] Lin, C.R., "On-demand QoS routing in multihop mobile networks," INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, pp. 1735-1744 vol. 3, 2001.

[6] S. Sriram, T.B. Reddy, B.S. Manoj, C.S.R. Murthy, "The Influence of QoS Routing on the Achievable Capacity in TDMA-Based Ad Hoc Wireless Networks", Proceedings of IEEE Global Telecommunications Conference (GLOBECOM), vol. 6, pp. 2909-2913, December 2004.

[7] Gore, A.D.; Karandikar, A.; Jagabathula, S., "On High Spatial Reuse Link Scheduling in STDMA Wireless Ad Hoc Networks," Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE, pp.736-741, 26-30 Nov. 2007.

[8] R. Tjoa, K.L Chee, P. K. Sivaprasad, S.V. Rao and J.G. Lim, "Clock Drift Reduction for Relative Time Slot TDMA-based Sensor Networks," in Proceedings of IEEE PIMRC, vol. 2, pp. 1042-1047, 2004.

[9] Yunbo Wang; Vuran, Mehmet C.; Goddard, S., "Cross-Layer Analysis of the End-to-End Delay Distribution in Wireless Sensor Networks," Networking, IEEE/ACM Transactions on , vol. 20, no. 1, pp. 305-318, Feb. 2012.

[10] V. Sevani, B. Raman and P. Joshi, "Implementation based evaluation of a full-fledged multi-hop TDMA-MAC for WiFi mesh networks", IEEE Trans. Mobile Comput., vol. 13, no. 2, pp. 392-406, 2014.

[11] Bei Wang; Bo Wang; Qingqing Xiong, "The comparison of communication methods between user and Kernel space in embedded Linux," Computational Problem-Solving (ICCP), 3-5 Dec. 2010.

[12] Vattikuti, N.; Sindhwal, H.; Dasari, M.; Reddy, T.B., "Delay Sensitive TDMA Slot Assignment in Ad Hoc Wireless Networks," Conference on Communications (NCC), Conference Proceedings, IEEE, 2015.