TCP CRAHN: A Transport Control Protocol for Cognitive Radio Ad Hoc Networks

Kaushik R. Chowdhury, *Member*, *IEEE*, Marco Di Felice, and Ian F. Akyildiz, *Fellow*, *IEEE*

Abstract—Cognitive Radio (CR) networks allow users to opportunistically transmit in the licensed spectrum bands, as long as the performance of the Primary Users (PUs) of the band is not degraded. Consequently, variation in spectrum availability with time and periodic spectrum sensing undertaken by the CR users have a pronounced effect on the higher layer protocol performance, such as at the transport layer. This paper investigates the limitations of classical TCP newReno in a CR ad hoc network environment, and proposes TCP CRAHN, a window-based TCP-friendly protocol. Our approach incorporates spectrum awareness by a combination of explicit feedback from the intermediate nodes and the destination. This is achieved by adapting the classical TCP rate control algorithm running at the source to closely interact with the physical layer channel information, the link layer functions of spectrum sensing and buffer management, and a predictive mobility framework that is developed at the network layer. An analysis of the expected throughput in TCP CRAHN is provided, and simulation results reveal significant improvements by using our approach. To the best of our knowledge, our approach takes the first steps toward the design of a transport layer for CR ad hoc networks.

Index Terms—Cognitive radio, congestion control, flow control, spectrum sensing, TCP

1 Introduction

The emerging field of Cognitive Radio (CR) networks attempts to alleviate the problem of spectrum scarcity in the ISM band by opportunistically transmitting on other vacant portions of the spectrum, such as frequencies licensed for television broadcast and public services [1].

In this paper, we consider CR Ad Hoc Networks (CRAHNs) that do not have a centralized entity for obtaining the spectrum usage information in the neighborhood, or external support in the form of a spectrum broker that enables the sharing of the available spectrum resource. Thus, compared to infrastructure-based networks, relying on local decisions makes the problem of node-coordination and end-to-end communication considerably more involved. While the mobility of the intermediate nodes and the inherent uncertainty in the wireless channel state are the key factors that affect the reliable end-to-end delivery of data in classical ad-hoc networks, several additional challenges exist in a CRAHN. The periodic spectrum sensing, channel switching operations, and the awareness of the activity of the Primary Users (PUs) are some of the features that must be integrated into the protocol design [1]. For these reasons, protocol development at the higher layers of the network stack for CR ad hoc networks, involving end-to-end communication over multiple hops, is still in a nascent stage. In this paper, we propose a window-based, TCP-like spectrum-aware transport layer protocol for CR ad-hoc networks, called TCP CRAHN that distinguishes between the different spectrum-specific conditions in order to undertake state-dependent recovery actions.

At the transport layer in classical wireless ad hoc networks, the main challenge lies in distinguishing 1) congestion, 2) channel-induced packet drops, and 3) mobility-based packet losses. In the first case, the packet experiences greater queuing delay in the buffers of the intermediate routes, thereby increasing the Round Trip Time (RTT). Consequently, TCP suffers from timeout events if the RTT exceeds a given threshold. In the second case of channel-related losses, such as those caused by fading or shadowing, the dropped packets are mistaken by the source as a congestion event. Mobility-related losses are mostly permanent, and if the sender already has a large number of in-flight packets, then all of them are likely to be lost. Though these loss-inducing factors are also applicable to CRAHNs, there are additional unique considerations: the observed RTT may increase if an intermediate node on the route is engaged in spectrum sensing and hence, unable to forward packets. Also, the sudden appearance of a licensed or primary user may force the CR nodes in its vicinity to cease their transmissions, leading to an increase in the RTT. In such cases, the network is partitioned until a new channel is identified and coordinated with the nodes on the path. As an example, in Fig. 1, consider a chain topology formed by the source S, destination D, and intermediate forwarding nodes. If the node 2 is performing spectrum sensing, then for that duration, it is unable to send or receive packets, resulting in a virtual disconnection of the path. Consequently, the data packets in node 1 and moving toward D, and acknowledgments (ACKs) in node 3 for the source S

- K.R. Chowdhury is with the Department of Electrical and Computer Engineering, Northeastern University, 409 Dana Research Center, Boston, MA 02115. E-mail: krc@ece.neu.edu.
- M.D. Felice is with the School of Computer Science, University of Bologna, Via M. Anteo Zamboni, Bologna 740126, Italy. E-mail: difelice@cs.unibo.it.
- I.F. Akyildiz is with the Broadband Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Centergy One, Room 5170, Atlanta, GA 30332. E-mail: ian@ece.gatech.edu.

Manuscript received 22 Dec. 2010; revised 24 Aug. 2011; accepted 14 Feb. 2012; published online 28 Feb. 2012.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2010-12-0585.
Digital Object Identifier no. 10.1109/TMC.2012.59.

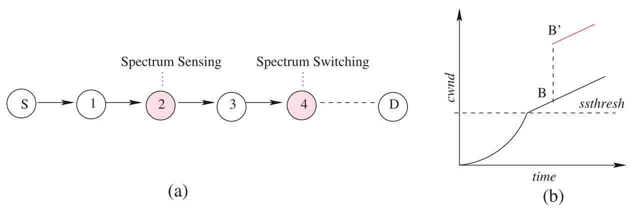


Fig. 1. (a) A multihop CR ad hoc network and (b) the forced *cwnd* scaling.

both experience greater queuing delays. If a timeout indeed occurs, the source is immediately penalized and the rate of sending data is drastically reduced. Similarly, consider the case in which the spectrum used by node 4 is reclaimed by the PUs, and it must immediately cease transmission. There is a finite time duration in which node 4 must identify a new spectrum, switch its transceivers, and coordinate this choice with its neighbors. Thus, in both the above cases of spectrum sensing and switching, the source may mistake the increased RTT (or timeouts caused by this increase) for congestion. In TCP CRAHN, we rely on the intermediate nodes periodically piggybacking their spectrum information on the ACKs, or in times of a sudden event like a PU arrival, explicitly notifying the source. While several works have focused on spectrum sensing algorithms in the last few years [1], the integration of the channel information collected at the nodes and the performance study of these approaches from the viewpoint of an end-to-end protocol remains an open challenge.

The local spectrum decisions undertaken by a node strongly influence the end-to-end performance. As an example, if the spectrum sensing duration is large, then the node can better detect the PU activity in its local area. However, this also results in lower end-to-end throughput [23]. Thus, the optimal balance between protection to the PUs (higher sensing time) with the increase in CR network throughput (lower sensing time) must also be decided, which is undertaken by TCP CRAHN. Moreover, our protocol accounts for the possibility that a channel switching event may result in a significant change in bandwidth of the affected link. Here, the number of packets that can be supported by the network in a unit time can suddenly increase, especially if the earlier spectrum was the *bottleneck* spectrum allowing very low date rates. Consequently, we propose an artificial scaling of the TCP congestion window (*cwnd*) to respond quickly to the change in the environment. As shown in Fig. 1b, when node 4 switches the spectrum, choosing a higher capacity channel for the link 4-5 (node 5 is not shown) the corresponding *cwnd* is increased immediately from its normal linear trajectory at B to a new value B′ that allows the source to fully utilize the spectrum. This is especially important as spectrum is available for limited durations, and the CR user must make the most efficient use of it. Our work also addresses several concerns of classical wireless ad hoc networks, such as the effect of mobility. By proactively predicting the possibility of a route outage through the method of Kalman filtering [12], the source can limit the number of unacknowledged packets at a given time, thereby also limiting the loss in the event of an actual route failure.

TCP, in general, is a well-researched area and several theoretical models exist that explain and predict its behavior in wireless networks [26]. It is also implemented at the transport layer for commercially available devices. In addition, the ad hoc network may ferry user traffic to and from the external infrastructure network, receiving configuration commands from remote stations. TCP is the defacto standard in the wired world and high compatibility with existing infrastructure is useful from the network management perspective. Hence, the goal of TCP CRAHN is to retain the window-based approach of the classical TCP, and at the same time introduce novel changes that allow its applicability in CR ad hoc networks. We would like to mention that the main merit of this paper lies in the theoretical design of a transport layer. The actual implementation on real software defined radios is currently limited by the lack of implementations for link layer and end to end network layer protocols. Thus, there are many practical issues that exist today, which make it difficult for demonstrating TCP CRAHN running on such radios, but we are hopeful that rapid advances will soon make this feasible. Note that our approach involves making several assumptions of the underlying protocol operation, which may not hold in practice in the eventual standardized implementations.

The rest of this paper is organized as follows: We give the related work in this area in Section 2. In Section 3, we motivate the need of a new transport protocol for CR networks. The network architecture is given in Section 4. In Section 5, we describe our transport layer protocol in detail, and provide an analysis of our method is Section 6. We undertake a thorough performance evaluation in Section 7, and finally, Section 8 concludes our work.

2 Related Work

Transport protocols constitute a well investigated topic in traditional wireless ad hoc networks, but they are quite unexplored in CR networks. It is well known that classical TCP implementations which run over the Internet (e.g., TCP newReno [10], TCP Vegas [3], and TCP SACK [20]) perform poorly over wireless links because of the additional packet losses caused by bad channel conditions or by node mobility, which are often misinterpreted as indicators of network congestion. As a result, several transport protocols have been proposed for wireless ad hoc networks, by using cross-layer or layered approach. As an example of the first approach, the Ad Hoc TCP Protocol (ATCP) [15] leverages network feedbacks from intermediate nodes to distinguish packet losses caused by mobility or by channel errors rather than by network congestion. Similarly, network feedbacks and cooperation among layers are also used by Sundaresan et al. [24]. In this layered approach, the source node does not rely on feedbacks from intermediate nodes, but it analyzes the correlation between TCP events (e.g., timeouts or out-of-delivery events) to detect the cause of a packet loss. This is also the case for the TCP-DOOR [25] and TCP-RTO [7] protocols. Moreover, there is a lot of the literature on how to enhance the performance of TCP by designing novel solutions at the MAC or link layers, without modifying the existing TCP standard [27]. However, we highlight that all these protocols are not suitable for CR

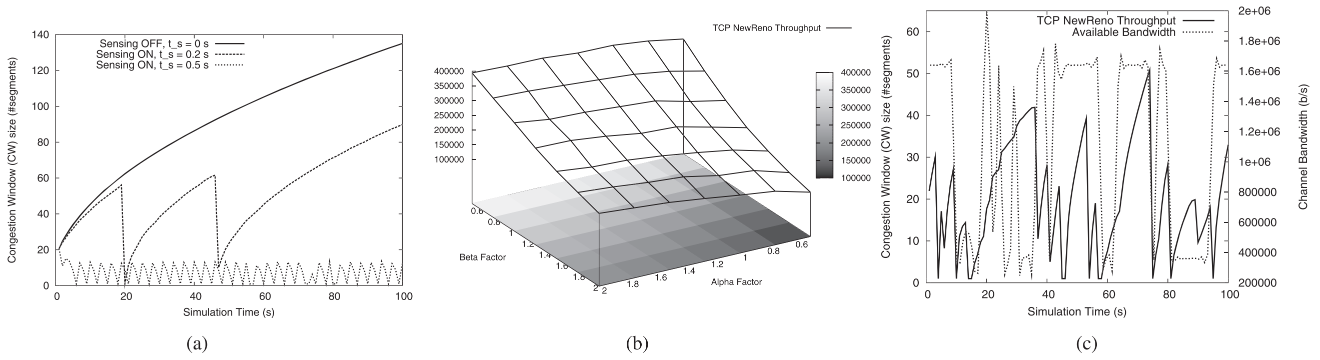


Fig. 2. (a) A study of the *cwnd* size as a function of the varying sensing time. (b) The impact of different PU activity on TCP throughput is investigated. (c) The effect of changing channel bandwidth on *cwnd*. All simulations are undertaken in ns-2.

networks, because they do not consider the special characteristics of CR networks. There are few works investigating the performance of classical TCP over CR networks, and even fewer addressing the design of novel transport protocol solutions. Slingerland et al. [23] and Kondareddy and Agrawal [13] provide insight on the effect of dynamic spectrum access (DSA) links over TCP performance, by using analytical and simulation tools. In both cases, the authors conclude that one dominating effect which is responsible for TCP throughput reduction is the sensing time on the current channel. In [16] and [17], Luo et al. describe a learning framework that decides several operational parameters, such as channel selection, sensing, channel access decision, modulation, coding scheme, and the frame size, using rewards measured on the basis of observed TCP throughput. However, this work does not propose any changes to the existing TCP newReno to respond to PU activity specific events. It is also a challenge to ensure that the learning algorithm can converge quickly in a CR ad hoc network that typically operates in a highly dynamic environment.

In [22], Sarkar and Narayan propose to modify the congestion window adaptation in the classical TCP Westwood [18] based on the available bandwidth estimation. Their approach prevents TCP from incorrectly reducing its transmission rate (or estimated bandwidth) during spectrum sensing. However, this work does not consider the impact of sensing activity, the durations for spectrum switching, and network mobility on the flow and congestion control functions. The Spectrum-Aware Event Transport (SET) suite for CR sensor networks is proposed in [2] that has a proactive congestion control algorithm, with a focus on energy-efficient and reliable delivery for both delay-insensitive and real-time data. However, SET does not provide packet-based reliability and optimize bandwidth utilization, as its objective is conserving energy. To the best of our knowledge, TCP CRAHN is the first work addressing all the above CR characteristics in an integrated manner, while retaining the TCP-like approach to enable communication over wired-wireless heterogeneous scenarios. The preliminary work on this protocol appeared in [6], which has been extended with improved motivation studies (Section 3), an analytical model for the purpose of comparison with classical TCP (Section 6), and new performance evaluation results (Section 7).

3 Motivation

In this section, we discuss the problems with the existing implementations of transport protocols based on TCP newReno in CR ad-hoc networks, in which, nodes are equipped with a single radio transceiver. The features of the CR network that we study are: 1) spectrum sensing 2) effect of primary user activity, and 3) spectrum change. On any given channel, the PU is modeled as Poisson arrivals, with an "*on*" time ($\frac{1}{\alpha}$) and "*off*" time ($\frac{1}{\beta}$).

3.1 Spectrum Sensing State

CR users periodically monitor the current channel over a pre-decided sensing duration for the occurrence of PUs before using it for transmission. During this interval, the nodes are not actively involved in transmitting data packets, and the multi-hop network is virtually disconnected at the node performing spectrum sensing.

In Fig. 2a, we show the impact of sensing-induced delay on TCP newReno performance, when there is no PU activity on the current channel. We analyze the behavior of the Congestion Window (CW) size under three different configurations of the sensing time t^s, i.e., 0 s (sensing disabled), 0.2 s, and 0.5 s. When sensing is disabled, we observe that the CW keeps increasing till the capacity of the channel is reached. When sensing is enabled, DATA and ACK packets experience an extra-delay which triggers timeout events at TCP sender side. As a result, TCP reduces the CW to 1 segment, and resets to the slow-start state. When t^s is equal to 0.5, the sensing delay is comparable with the maximum retransmission timeout (RTO) timer value, and thus, frequent RTO events are triggered, degrading the end-to-end performance. This analysis is also in accordance with results shown in [22]. In [6], we also showed that the duration of t^s can play a critical role in deciding the optimal end-to-end throughput, because it constitutes a trade-off between 1) accurate PU detection and 2) efficient channel utilization. Thus, it is responsibility of the transport layer to adapt the current rate during the sensing state, and to decide the optimal setting of t^s so that the throughput is maintained at the desired level while the interference on PUs is minimized.

3.2 Effect of PU Activity

On detecting the presence of a PU, either during spectrum sensing or an ongoing data transfer, the CR users cease their operation on the affected channel and search for a different vacant portion of the spectrum. While the spectrum sensing on the current channel is periodic and has a well defined interval, the time taken to 1) search for a set of available channels on different spectrum bands, and 2) coordinate with the next hop neighbors to find a mutually acceptable channel in this set, is generally uncertain. Moreover, the path to the destination is disconnected until the new channel is successfully found, the time for which is not known to the source in advance. Thus, the transport protocol needs to differentiate this state from other causes of route disconnections with the help of an explicit feedback from the nodes affected by the PU activity.

In Fig. 2b, we show the impact of PU activity in terms of average *"on"*-time (x-axis) and *"off"*-time (y-axis) on the TCP newReno throughout (z-axis). Based on the values of α and β, it is possible to distinguish among four different patterns of PU activity: High-Activity Region ($\frac{1}{\alpha} \leq 1, \frac{1}{\beta} > 1$), Low-Activity Region ($\frac{1}{\alpha} > 1, \frac{1}{\beta} \leq 1$), Short-Term Activity region ($\frac{1}{\alpha} > 1, \frac{1}{\beta} > 1$), and Long-Term Activity region ($\frac{1}{\alpha} \leq 1, \frac{1}{\beta} \leq 1$). Not surprisingly, TCP performance is maximized when the CRs have more possibility to access the licensed spectrum without interfering with the PU activity, (i.e., in the Low Activity Region) and minimized when the PUs are more active on the current channel (i.e., High Activity Region). At the same time, results shown in Fig. 2b and discussed in [8] demonstrate that TCP suffers of performance decrease when there are frequent *"on"*-*"off"* switches (i.e., in the Short Term Activity Region) due to the fact that the CW can not increase because of frequent PU arrivals on the current channel. As a result, we believe that transport layer should be informed of spectrum handoff operations occurring at the lower layer, in order to distinguish packet losses caused by congestion or by PU interference.

3.3 Spectrum Change State

A key concern in CR networks is the efficient utilization of the spectrum resource, as the opportunity for transmission in the licensed bands is available for a limited time. The licensed channels may have a large variation in bandwidth, especially as nodes switch from one spectrum band to the other. In Fig. 2c, we study through simulation how classical TCP increases the *cwnd* as it probes for the additional bandwidth available on a single link. There are three different channel bandwidths possible—2/3 Mbps, 4/3 Mbps, and 2 Mbps. The vertical bars denote the bandwidth available to the node and at any given time, this is the upper limit that can be utilized by the TCP connection. This gives three distinct levels of bandwidth availability with time. On each channel, the PU is modeled as a Poisson arrival, with an *"on"* time ($\frac{1}{\alpha} = 4$ s) and *"off"* time ($\frac{1}{\beta} = 5$ s). When the PU arrives, the CR user switches to a different channel, and consequently TCP must adjust to the new available bandwidth. From the figure, we observe that the *cwnd* is unable to correctly track the available bandwidth. Moreover, the spectrum opportunity is often lost before the *cwnd* has increased to half the segments that may be supported on the new channel. A similar conclusion is drawn in [23], where TCP cannot effectively adapt to brief reductions in capacity, if the end-to-end delay is large. We believe that the *cwnd* in TCP must be scaled appropriately to meet the new channel conditions, as shown in the transition from the operating point B to the point B′ in Fig. 1b. Estimating this new operating point is a challenge and link layer metrics that determine the effective bandwidth, must also be considered apart from the raw bandwidth. Bandwidth estimation techniques have been proposed in [4] and [18] that do not require information from the intermediate nodes, but also do not respond immediately to the available spectrum.

4 Network Architecture

The nodes forming the CR ad hoc network have a single radio transceiver that can be tuned to any channel in the licensed spectrum. We assume ψ spectrum bands are present with $n(x)$ channels in a given band x. The channels of this spectrum band are denoted by $\xi_p^x, \;\; p = 1, \ldots, n(x)$. In general, two channels in different spectrum bands may have dissimilar raw channel bandwidth, i.e., $\xi_p^x \neq \xi_q^y$. In addition, we assume the statistical knowledge of the PU arrival (α) and departure rate (β) for each channel are known, so that an initial estimate of the channel sensing time can be calculated.

We use CSMA/CA at the Medium Access Control (MAC) layer that has a predecided Common Control Channel (CCC) for coordination of the spectrum band and channel during data transfer. We also use a priority queue, Q_p at the MAC layer for the TCP CRAHN control packets, which may also be drawn from intermediate positions in Q_p.

In a CR network, nodes maintain a list of unoccupied channels (other than the current one in use) that may belong to different spectrum bands. In our work, we assume that this set of channels is identified through spectrum sensing, undertaken during the *backoff* interval following a packet transmission or reception at the link layer. On the current operational channel, however, it is important to have an accurate idea of the PU activity. For this, we do not rely on probabilistic sensing times. Rather, nodes sense their current channel for the sensing time t^s at regular intervals at the cost of continued network connection [14].

5 TCP CRAHN: A Transport Protocol for CR Ad Hoc Networks

TCP CRAHN comprises of the following 6 states, as shown by the state diagram in Fig. 3. They are

1. Connection establishment,
2. Normal,
3. Spectrum sensing,
4. Spectrum change,
5. Mobility predicted, and
6. Route failure.

Each of these states addresses a particular CR network condition and we describe them in detail as follows.

5.1 Connection Establishment

TCP CRAHN modifies the three-way handshake in TCP newReno so that the source can obtain the sensing schedules of the nodes in the routing path. First, the source

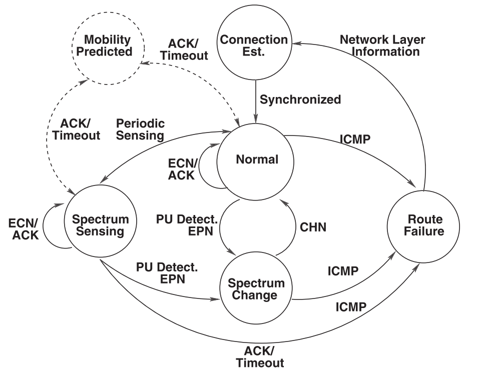


Fig. 3. Finite state machine model of TCP CRAHN.

sends out a Synchronization (SYN) packet to the destination. An intermediate node, say i, in the routing path appends the following information to the SYN packet: 1) its ID, 2) a timestamp, and 3) the tuple $\{t_i^1, t_i^2, t_i^s\}$. Here, t_i^1 is the time left before the node starts the next round of spectrum sensing, measured from the timestamp. t_i^2 is the constant duration between two successive spectrum sensing events, and t_i^s is the time taken to complete the sensing in the current cycle. On receiving the SYN packet, the receiver sends a SYN-ACK message to the source. The sensing information collected for each node is piggybacked over the SYN-ACK and thus, the source knows when a node in the path shall undertake spectrum sensing and its duration. The final ACK is then sent by the source to the destination completing the handshake.

We note that the calculation of the sensing time t_i^s by a node i is undertaken locally. Based on the bandwidth of the channel (W), the external signal to noise ratio (γ), and the probabilities of the on period (P_{on}) and the off period (P_{off}), a framework to calculate this time is given as follows [14]:

$$t_i^s = \frac{1}{W\gamma^2}\left[Q^{-1}(P_f) + (\gamma+1)Q^{-1}\left(\frac{P_{off}P_f}{P_{on}}\right)\right]^2, \qquad (1)$$

Equation (1) gives the sensing time t_i^s that minimizes the probability of missed primary user detection P_f, i.e., incorrectly stating the channel is vacant when indeed there is an active PU and Q is the standard Q function. The sensing times collected from the nodes are the preliminary values which are dynamically updated by TCP CRAHN, as described in Section 5.3.

State transitions. On successful handshake, the source and destination are synchronized and the *Normal* state is entered.

5.2 Normal State

The *normal* state in TCP CRAHN is the default state and resembles the classical functioning of the classical TCP newReno protocol. Our protocol enters this state when 1) no node in the path is currently engaged in spectrum sensing, 2) there are no connection breaks due to PU arrivals, and 3) no impending route failure is signaled. Thus, the path to the destination remains connected and ACKs sent by the latter are received at the source. The differences between TCP CRAHN in the *normal* state and the classical TCP are as follows.

5.2.1 Explicit Congestion Notification

The congestion control algorithm in classical TCP operates in two phases, namely, the *slow start* and the *congestion avoidance*. In the slow start, the congestion window *cwnd* is initially set to 1 (Maximum Size Segment (MSS)) and doubled for each incoming ACK. As TCP probes for the available bandwidth, the *cwnd* increases exponentially until the threshold, *ssthresh*, is reached. It then enters the collision avoidance phase, where the *cwnd* is incremented by 1 MSS for every acknowledged packet. During network congestion, indicated by the retransmission timeout events, TCP reduces the *cwnd* to 1 and the new threshold is set to the value $\frac{ssthresh}{2}$.

While the above ACK based self-clocking mechanism that increases the *cwnd* is retained in TCP CRAHN, the congestion event is signaled through an Explicit Feedback Congestion Notification (ECN) generated by the affected node. The concept of explicit notification is not new, and prior works such as ATCP [15] and TCP-EFLN [11] rely on the source being informed of route outages through Internet Control Message Protocol (ICMP) messages at the IP layer. In TCP CRAHN, the congestion is detected at the node by comparing the current buffer usage for the given flow with a predecided threshold value B_{con}^f. The ECN is sent in two ways to guarantee its timely delivery. First, a packet is sent from the affected node to the source directly. In addition, the ECN is piggybacked to the destination over the data packets and then sent to the source through the ACK. This is done as the remainder of the connection from the affected node to the destination may suffer delay from a temporary disruption caused by channel sensing or switching. When an ECN is received by the source, TCP CRAHN first evaluates if it is still relevant to the network congestion state by checking the time lag from its generation at the affected node to its reception. If this time is within the time lag threshold L_{max} and no prior action has been taken for an earlier ECN from the same node, for the detected congestion event, TCP CRAHN reduces the *cwnd* to 1 and cuts the *ssthresh* by half. In our work, we set $L_{max} = 1.5 \times RTT$, as any further delay suggests that the path was temporarily disconnected due to a sensing or channel switching event. In either case, the transmission rate at the source is reduced, as we shall see later in the protocol description.

5.2.2 Feedback Through the ACK

The intermediate nodes of the path piggyback the following link-layer information over the data packets to the destination, which is then sent to the source through ACKs.

- *Residual buffer space* (B_i^f). Consider a node i that has B_i^u unoccupied buffer space. Let the number of flows passing through it be n_i^f. The fair share of the residual buffer space per flow is, $B_i^f = \frac{B_i^u}{n_i^f}$.
- *Observed link bandwidth* ($W_{i,i+1}$). Each node i maintains a weighted average of the observed bandwidth on the link formed with its next hop, i.e., $\{i, i+1\}$, during the *normal* state. This is obtained from the link layer as the ratio of the acknowledged data bits to the time taken for this transfer between the nodes i and $i+1$.

- *Total link latency* ($L^T_{i,i+1}$). Let $L_{i,i+1}$ be the sum of the 1) time taken by a packet of the current flow to move to the head of the queue 2) the time for contending the access to the channel and finally, 3) the transmission time measured at node i with respect to the next hop $i+1$. The total bidirectional link latency is hence $L^T_{i,i+1} = L_{i,i+1} + L_{i+1,i}$.

Apart from these fields that are updated every time by the nodes, the ACK also carries the ECN notification whenever a node experiences congestion and the Mobility predicted Flag (MF), which is set when there is a possibility of route disconnection (explained in detail in Section 5.5).

State transitions. The ECN message and the ACKs regulate the *cwnd*. When a node in the path performs sensing, TCP CRAHN transitions into the *Spectrum Sensing* state. If PU activity is reported to the source through an Explicit Pause Notification (EPN), it enters the *Spectrum Change* state and resumes the usual operation on receiving the information about the new channel through the Channel (CHN) message. Possible route disruptions may be signaled by the ACKs leading to the temporary *Mobility Predicted* state, where the *cwnd* is restricted to *ssthresh*. If the ICMP message is received, TCP CRAHN enters into the *Route Failure* state and stops transmission.

5.3 Spectrum Sensing State

We describe how TCP CRAHN adapts to spectrum sensing through 1) flow control, which prevents buffer overflow for the intermediate nodes during sensing and 2) regulating the sensing time to meet the specified throughput demands.

5.3.1 Flow Control

When a node i undertakes spectrum sensing, the path gets virtually disconnected for a finite duration. At this time, the goal of TCP CRAHN is to adapt the flow control mechanism in TCP, so that the node $i-1$, prior to the sensing node, is not overwhelmed with incoming data packets. If another node j has an overlapping sensing schedule, TCP CRAHN uses the residual buffer space of the previous hop of the node closest to the source during the period of overlap, say i. When the sensing time of the closest node is completed, the buffer space of node $j-1$ is used in the *ewnd* computations.

We recall that, in classical TCP, the maximum number of bytes of unacknowledged data allowed at the sender is the minimum of the current congestion window, *cwnd*, and the receive window advertised by the destination, *rwind*. The *rwind* represents the free space in the receiver's buffer that can accommodate additional transmitted packets. During the sensing duration, no ACKs are received by the source and hence the *rwind* remains unchanged. This also results in a constant *cwnd* as TCP is self clocked and does not increase in the absence of the receiver ACK. The effective window, *ewnd* at the sender is modified to include an estimate of the free buffer space, B^f_{i-1}, at the previous hop node $i-1$ as $ewnd = \min\{cwnd, rwnd, B^f_{i-1}\}$.

As the packets fill up the buffer in the node $i-1$, the remaining free buffer space needs to be progressively reduced. The intermediate node, unlike the destination, does not send back the ACKs with the new advertised receive window and hence, this is estimated as follows: from

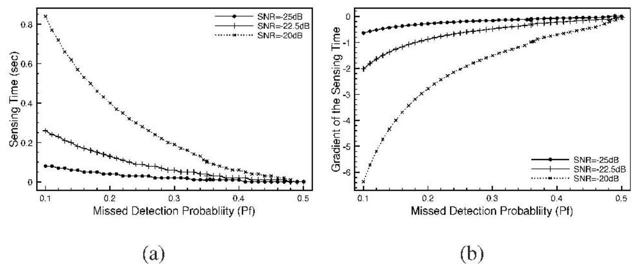


Fig. 4. The sensing duration for increasing error probability and the gradient of the curves are shown in (a) and (b), respectively, as obtained from (1). We assume $\alpha = 0.5$, $\beta = 0.25$, giving $P_{on} = 0.333$ and $P_{off} = 0.67$.

the last received ACK, we know the free buffer space for the given flow, at the node $i-1$, is B^f_{i-1}. The approximate time for successfully transmitting a packet over the link $i-2, i-1$ can be calculated at the source as

$$L_{i-2,i-1} = \frac{L^T_{i-1,i-2}}{2},$$

where $L^T_{i-1,i-2}$ is the bidirectional link latency piggybacked over the ACK. Thus, the space available B^f_{i-1} in the node $i-1$ is decremented at intervals of $L_{i-2,i-1}$, when node i is engaged in sensing. We note that while the rate of decrease of the buffer space is not exact, the node $i-1$ is oblivious to this sender-side adaptation. It can still force the source to reduce its sending rate through the congestion notification. If its buffer is reaching the overflow limit, the congestion condition will be signaled and the *cwnd* will be reduced to 1 at the source.

If any of the intermediate nodes on the path from the source to the node $i-1$ detect congestion, the ECN packet is sent by them and the *cwnd* is then reduced to 1. We note that the effective window *ewnd* remains at 1, as long as the path remains disconnected, as the *cwnd* cannot be increased without the ACK.

5.3.2 Sensing Time Regulation

In Section 3.1, we stated that if there is no PU activity on a given channel, the comparatively large sensing times degrade the end-to-end throughput. To address this, TCP CRAHN conservatively reduces t^s for the nodes that see limited PU activity. This calculation is carried out at the transport layer as it is aware of the observed (τ_o) and the desired (τ_d) throughputs, respectively. The two inputs needed by our protocol are—1) the value, δ^s_i, by which the sensing time should be decreased, and 2) the node i at which this reduction must be undertaken.

Sensing time decrease. Figs. 4a and 4b give the optimal sensing time and the gradient of the curve for the sensing time, respectively, in order to maintain a given missed detection probability (P_f) for different SNR ranges and a bandwidth of 2 MHz, as per the analytical formulation (1) from [14].

We observe that the t^s is large when the target error probability is very low. Moreover, there is also a large fall in the t^s for a finite change in the P_f, as shown by the gradient curve (Fig. 4b), when the P_f is small. This means that the t^s can be reduced by a greater margin in the initial stage, when it has a comparatively higher duration, without impacting the error significantly. The intuitive reasoning is as follows: a node i first sets $t^s = t^s_{max}$ corresponding to the low error probability of $P_f = 0.1$. If the number of spectrum changes that occur over time is small in proportion to the number of total changes, we assume that the node is situated in a region with limited PU activity. Thus, the periodic sensing time t^s_i may be reduced at this node. If the current t^s_i at the node is large, its reduction is consequently higher, as the probability of error is not affected proportionally. However, as the t^s_i value falls, the reduction gets progressively smaller until $t^s_i = t^s_{min}$ is reached, corresponding to the limiting error probability, $P_f = 0.5$. We can now formulate the steps for obtaining the new sensing duration $t^s_i(\text{new})$ from the old value t^s_i as follows:

$$\triangle_t = \frac{t^s_{max}}{2}, \tag{2}$$

$$\gamma_{t^s_i} = \frac{\mathrm{d}t^s}{\mathrm{d}P_f}|_{t^s=t^s_i,\ P_f=func(t^s_i)}, \tag{3}$$

$$\gamma_{t^s_{max}} = \frac{\mathrm{d}t^s}{\mathrm{d}P_f}|_{t^s=t^s_{max},\ P_f=0.1}, \tag{4}$$

$$\delta^s_i = -\frac{\gamma_{t^s_i}}{\gamma_{t^s_{max}}}\triangle_t, \tag{5}$$

$$t^s_i(\text{new}) = t^s_i - \delta^s_i. \tag{6}$$

The default decrement value of the sensing time, $\triangle_t$, is taken as half of the maximum value, t^s_{max}, needed to maintain the error probability at 0.1, as shown in (3). This is later scaled by a factor in the range [0,1] to get the true decrement δ^s_i. In (4), we calculate the value of the gradient $\gamma_{t^s_i}$ to the sensing curve at the current sensing duration t^s_i, at node i. The corresponding value of P_f is obtained from the current t^s_i from Fig. 4a, which is in turn, a numerical plot of (1). The maximum gradient of the sensing curve $\gamma_{t^s_{max}}$ is given in (5) and is computed at $t^s_i = t^s_{max}$. The normalized gradient, $\frac{\gamma_{t^s_i}}{\gamma_{t^s_{max}}}$, at the current tuple given by $\{t^s_i, P_f\}$ is used as the scaling factor to give the true decrement δ^s_i in (6). Finally, the sensing time is adjusted to the new value $t^s_i(\text{new})$ in (7).

When successive missed detection events occur, the node increases the sensing duration in the steps $\{\frac{1}{2} \times t^s_{max}, \frac{3}{4} \times t^s_{max}, t^s_{max}\}$, in that order. Whenever the sensing time is changed, the node sends back the new value to the sources of the flows passing through it by piggybacking over the ACK.

Node selection. In order to identify a specific node i for adjusting the sensing time, TCP CRAHN ranks the nodes in the path based on the number of times the operational channel was changed due to PU activity. It keeps a count of the CHN messages sent by each node of the path, which reveals the number of times the connection was paused while a new channel was being coordinated. Intuitively, the node that generated the highest proportion of the CHN message also experienced the maximum number of PU detection events and thus, must be located in a region of frequent PU activity. Such a node needs to retain a higher sensing duration.

Let the total number of times the spectrum change occurs at a given node i, and that considering all the nodes of the path be given by η_i and η_T, respectively. We define the probability of the node i being susceptible to PU activity, S_i as the ratio $S_i = \frac{\eta_i}{\eta_T}$. Let the set of n nodes along the route have their PU activity susceptibility given by the set $\mathbf{S} = \{S_1, \ldots, S_n\}$. Recalling that τ_d and τ_o are the desired and observed throughputs, the source executes the following algorithm to determine the node q and adjust its sensing time to the new value $t^s_q(\text{new})$:

```
PROCEDURE:Sense-Adjust

Input: $\tau_d$, $\tau_o$, S
Output: $q$, $t^s_q(\text{new})$
if $\tau_d > \tau_o$ then
    $q = \arg_i \min\{S_i\},\ i = 1, \ldots, n$
    if $t^s_q(\text{old}) > t^s_{min}$ & $S_i < S_{max}$ then
        $t^s_q(\text{new}) = t^s_q(\text{old}) - \delta^s_i$
    end
end
```

We explain the algorithm as follows: if the desired throughput (τ_d) is greater than the observed throughput (τ_o), then TCP CRAHN finds the node q with the minimum PU susceptibility, $S_i,\ i = 1, \ldots, n.$ Two conditions are checked for the node q—1) the current sensing time at the node t^s_q must be greater than the minimum allowed sensing time t^s_{min} and 2), the PU susceptibility must be below the limiting threshold s_{max}, considered as 0.3 in our work. This ensures that only nodes that are relatively undisturbed by PU activity over time are chosen for reduction of the sensing duration by the value δ^s_i, as described in (7). This new value of the sensing interval is sent to the intermediate node by the source.

State transitions. On receiving the EPN message, the *Sensing* state is interrupted and our protocol immediately transitions to the *Spectrum Change* state, or else, it reverts back to the *Normal* state on the completion of the sensing duration. The transitions to the *Mobility Predicted* and the *Route Failure* state are similar to the description of the *Normal* state.

5.4 Spectrum Change State

In the ideal case, the effective bandwidth of the TCP connection is dependent on several factors, such as contention delays and channel errors at the link layer, apart from the raw bandwidth of the channel. In this section, we show how TCP CRAHN scales its *cwnd* rapidly, say from point B to a different value B′, in Fig. 1b, accounting for these factors, so that the available spectrum resource is most efficiently utilized.

Consider three nodes given by $i-1$, i, and $i+1$ on the current path and the channels used by the links $\{i-1, i\}$ and $\{i, i+1\}$ be $c_{i-1,i}$ and $c_{i,i+1}$, respectively (Fig. 5a). If the PU is on the channel ξ^x_p and either $c_{i-1,i} = \xi^x_p$ or $c_{i,i+1} = \xi^x_p$, the node i must search for a new channel to prevent interference to itself and to the PU, respectively. At this stage, it sends an explicit pause notification to the source, which in turn, freezes the protocol state and waits for a new channel CHN message to resume the transmission. We consider the case where TCP CRAHN adjusts to a single

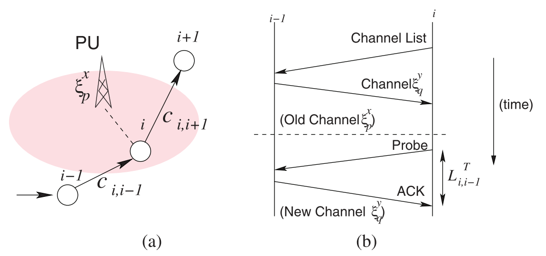


Fig. 5. (a) The PU interference scenario and (b) the link layer total delay estimation are shown.

affected link $\{i-1, i\}$ and then extend the analysis for the case when both the previous and next hop links need a channel change.

The set of available channels is known at node i, as described in Section 4. The preferred list of channels, from this available set, is sent by this node to the previous hop $i-1$ (Fig. 5b). The node $i-1$ chooses a channel from this set, say ξ_q^x. It then sends back a link layer ACK to node i to inform the node of its choice, ξ_q^x. All the coordination up to this point occurs on the old channel. A second set of Probe and ACK messages are then exchanged on the channel to be switched, ξ_q^x, as a confirmation and also to approximately estimate the new link transmission delay times $L_{i,i-1}$ and $L_{i-1,i}$. If the probe and ACK packets are of the size P_{probe} and P_{ACK}, respectively, the observed link bandwidth $W_{i,i-1}$ is

$$W_{i,i-1} = \frac{P_{probe} + P_{ACK}}{L_{i,i-1} + L_{i-1,i}}. \tag{7}$$

The CHN message contains in it the bidirectional link layer packet delay over the newly identified channel, ($L_{i,i-1}^T = L_{i,i-1} + L_{i-1,i}$) that is used by the source to calculate $W_{i,i-1}$ from (7). From Section 5.2.2, we recall that the ACKs forwarded over the intermediate hops also carry the total bidirectional link latency, $L'^T_{i,i-1}$, corresponding to the earlier used channel. On receiving the CHN message, the source first estimates the new RTT using 1) the earlier observed RTT′ during the last *normal* state of the protocol and 2) adjusting for the new bidirectional link delay, $L_{i,i-1}^T$. This is given by $RTT = RTT' + L_{i,i-1}^T - L'^T_{i,i-1}$.

For the given path of n nodes, let W_b' be the old observed bottleneck bandwidth, before the channel change. After the channel change, the new bottleneck bandwidth is identified as W_b, where $W_b = \min\{W_{l,l+1}\}$, $l = 1, \ldots, n-1$. The updated estimate of the bandwidth $W_{i,i-1}$ is used in this calculation from (7). If the ratio of the old bottleneck bandwidth to the new is within the allowed range of $[1-\varpi, 1+\varpi]$, i.e., $\frac{W_b'}{W_b} \in [1-\varpi, 1+\varpi]$, then no scaling of the earlier *cwnd* is needed, where $\varpi = 0.2$. If it lies beyond this range, then we calculate the new value of the *cwnd* as follows:

$$cwnd = \alpha_c W_b RTT. \tag{8}$$

The factor $\alpha_c = 0.8$ in (8) is used to adjust the *cwnd* to a value slightly lower than the predicted bandwidth to prevent the risk of overestimating the *cwnd* [5]. Over time, the *cwnd* converges to the optimal value around this range. In the event that the channels of both the upstream and downstream links are changed, the bidirectional link latencies, $L_{i,i-1}^T$ and $L_{i,i+1}^T$ are used in (7) and (8).

State transitions. The *Spectrum Change* state is entered as soon as an EPN message is received. It reverts back to the *Normal* state when the new channel information is received in the CHN message or enters into the *Route Failure* state on the receipt of an ICMP message. Existing sensing schedules are ignored as long as the protocol stays in the current state.

5.5 Mobility Predicted State

In order to address the problem of delayed route failure notification (Section 3), we develop a mobility prediction framework based on Kalman filter-based estimation [12], which uses the Received Signal Strength (RSS) information from the link layer. We construct the set of Kalman equations similar to the disposition for calculating sensor location in [19], but for a simpler, scalar case of a single dimension of the received power value. Formally,

$$x^k = \mathbf{F}x^{k-1} + \mathbf{G}u^{k-1} + \mathbf{B}w^{k-1}, \tag{9}$$

describes the transition between the states for the system used for predicting the new RSS value (i.e., x^k) at the kth iteration, using the previous prediction (i.e., x^{k-1}), and current control input or observed RSS value (i.e., u^{k-1}), and the variable w^k represents discrete random changes owing to fading or multipath. This 1D sample $w^k \sim N(0, Q)$, where $Q \geq 0$ is the covariance value of the process. Furthermore, the gains $\mathbf{F} = \mathbf{G} = \mathbf{B} = 1$, as described in [19]. The remaining steps of updating the Kalman filter gain, calculating the new value of the prediction, i.e., the estimate of the RSS x^k, and updating the covariance value of the obtained samples follow the classical steps described in [19] and are not replicated here.

The nodes of the path monitor the connectivity to their next hop downstream node by measuring the RSS of the ACKs and the periodic beacon messages. At each epoch, the prediction value is compared with the minimum RSS required for receiver operation. If the condition of possible link failure is predicted in the next epoch, the destination is informed, which then sets the *Mobility Flag* (MF) in the outgoing ACKs. The source responds to this by limiting the *cwnd* to the *ssthresh* and the congestion avoidance phase is never initiated. The aim of this adjustment, cwnd $\leq$ *ssthresh*, is to limit the number of packets injected into the route which has a possibility of an outage, as the CR specific function of the nodes may delay the arrival of the actual link failure notification. If no ICMP message is received at the source subsequently, signaling that a route failure has indeed occurred or the incoming ACKs do not have the MF flag sent, the mobility prediction state is cancelled and TCP CRAHN reverts back to the *normal* state, where the *cwnd* is no longer bounded.

State transitions. TCP CRAHN regards the *Mobility Predicted* state as a transient or virtual state, in which the *cwnd* is restricted to the *ssthresh* and the current operation either in the *Normal* or the *Spectrum Sensing* state is continued.

5.6 Route Failure State

The node i sends a *destination unreachable* message in the form of an ICMP packet if 1) the next hop node $i+1$ is not reachable based on link layer retries, 2) there is no ongoing spectrum sensing based on the last known schedule, and 3) no EPN message is received at node i signaling a temporary path disconnection due to PU activity. At this stage, the source stops transmission and a fresh connection needs to be formed over the new route by TCP CRAHN.

State transitions. The *Route Failure* state is the terminal state of the current cycle and a fresh TCP connection must be established when a new route is formed. The protocol enters this state on receiving the ICMP message and it takes precedence over all the others states.

6 TCP CRAHN Throughput Analysis

In this section, we derive an analytical expression for throughput in TCP CRAHN, using a previously existing model for TCP newReno. Similar to other modeling efforts [21], we ignore the slow-start phase, thereby allowing us to view the *cwnd* as a concatenation of statistically independent cycles. Thus, in each cycle, the *cwnd* increases linearly till a loss event occurs, resulting in the immediate reduction of the *cwnd* to half, followed by fast recovery. To keep the analysis tractable, we assume that the spectrum switches do not result in large scale bandwidth availability changes (i.e., *cwnd* scaling derived in (8) is absent).

Consider a chain topology of k Cognitive Radio nodes with an active TCP connection between the end-points, and M licensed channels. Each channel can be occupied by a PU which follows an alternate *on-off* activity pattern with the mean durations represented by $\frac{1}{\alpha}$ and $\frac{1}{\beta}$, respectively. Each CR node performs sensing for t^s time units and transmission for T_p time units, alternately. We assume that time is divided into logical slots. Our analysis focuses on the *normal*, *spectrum sensing*, and *spectrum switching* states of the TCP CRAHN protocol, and in any given slot, the protocol can be in one of these states. In summary,

- *Normal* state: for a duration T_p. This represents the normal situation in which the network is connected and CR can transmit data packets.
- *Spectrum sensing* state: for a duration t^s. There is at least one CR node of the chain which is sensing channel and thus it is unable to transmit data (network is disconnected).
- *Spectrum change* state. There is at least one CR node of the chain which is switching channel because of PU interference on the current channel (network is disconnected) for a time duration t_c.

Based on the duration and frequency of each logical slot, the TCP CRAHN throughput (B) in each time slot can be derived as follows:

$$B = \frac{B_n(1 - P_s - P_c)T_p + B_s P_s t^s + B_c P_c t_c}{P_s t^s + P_c t_c + (1 - P_s - P_c)T_p}, \tag{10}$$

where

- P_s: probability that the network is in the spectrum sensing state.
- B_s: TCP-throughput in the spectrum sensing state.
- P_c: probability that the network is in the spectrum change state.
- B_c: TCP-throughput in the spectrum change state.
- T_c: spectrum handoff delay. It includes channel switching delay, overhead for protocol reconfiguration (at layers 2/3), negotiation of a new channel with the next-hop node, etc.
- P_N: probability that the network is in the normal spectrum state.
- B_n: TCP-throughput in the normal spectrum state.

The first term in the numerator gives the effective number of bits transmitted in the normal state. For this, the protocol has to first be in the *normal* state, given by the probability $(1 - P_s - P_c)$, i.e., not in the *spectrum sensing* or the *spectrum change* states. The effective time for which TCP CRAHN enjoys the *normal* state bandwidth B_n is hence $B_n(1 - P_s - P_c)T_p$. Similarly, the second and the third terms in the numerator gives the bits transmitted in the *spectrum sensing* state, and the *spectrum change* state, respectively. This separate consideration of the individual states is the key characteristics of TCP CRAHN, and classical TCP is unaware of the distinction of B_n, B_s, and B_c in the absence of cross-layer and node-level feedback. We now derive the expressions for the variables in (10), B_n, to arrive at the closed form analytical expression.

6.1 *Normal* State Analysis

In the *Normal* state, the TCP CRAHN source follows the congestion control of classical TCP, and thus the achievable throughput B_n can be derived by using of the many existing TCP analytical models in the literature [21], [23]. In our case, we use the SQRT model, which provides a simple yet effective model for estimating the TCP throughput [23] in the normal state, as follows:

$$B_n = \frac{DATA}{RTT}\sqrt{\frac{3}{2p}}, \tag{11}$$

where $DATA$ is the size of the TCP segment, RTT is the round trip time for data delivery and receiving the confirmation, and p is the probability of end-to-end packet loss.

In our case, the RTT is the average time to deliver a TCP-DATA message to the destination, and receive an TCP-ACK message over a network chain of k nodes. At the link layer, assuming a CSMA/CA protocol with ARQ and backoff procedure, the average RTT can be approximated as $RTT = \overline{n}k(T_{DATA} + T_{ACK})$, where $\overline{n}$ is the average number of MAC re-transmission experienced on each link of the chain. T_{DATA} and T_{ACK} are the time required to transmit a TCP data and a TCP-ACK,

Now, $\overline{n}$ can be calculated as $\overline{n} = \sum_{i=0}^{L-1} i p_e^i (1 - p_e)$, where L is the maximum number of retries defined in the CSMA/CA protocol (e.g., 7 in the MAC DCF 802.11 standard) and p_e is the probability of packet loss at link-layer. We consider two causes of packet loss in the network: 1) transmission errors due to channel effects (e.g., shadowing; modeled as a uniform distribution with probability p_T) and 2) interference from PU. We also assume that these two causes are independent, and do not occur simultaneously. Thus, we get

$$p_e = p_I + p_T, \tag{12}$$

where p_I is the probability of packet loss due to PU interference. This is a function of the activity of the PU on the current channel as well as of the probability of correct detection (p_d). The probability of correct detection (p_d) is a function of α, β and t^s, and can be derived from [14]:

$$p_d = \frac{\beta}{\alpha+\beta} Q\left(\frac{\lambda - 2t^s W(\sigma_s^2+\sigma_n^2)}{\sqrt{4t^s W(\sigma_s^2+\sigma_n^2)^2}}\right). \tag{13}$$

Thus, the probability of PU induced packet loss (p_I) is the product of the probability of PU being in the "on' state ($\frac{\beta}{\alpha+\beta}$), and the probability of missed detection ($1-p_d$). Hence,

$$p_I = \frac{\beta}{\alpha+\beta}(1-p_d). \tag{14}$$

For the calculation of p_T, we assume a Rayleigh fading channel and M-PSK modulation (currently used in IEEE 802.11b systems). The Bit Error Rate (BER) is

$$BER = 1 - \sqrt{\frac{m'.r_c.\gamma \sin^2\frac{\pi}{M}}{1+m'.r_c.\gamma \sin^2\frac{\pi}{M}}}$$

[28], where, $M = 2^{m'}$ gives the order of modulation and r_c is the rate encoder used. The probability of the packet error assuming a single bit error is irrecoverable is[1]

$$p_T = 1-(1-BER)^{DATA}. \tag{15}$$

After calculating p_e from (12), (14), and (15), we can calculate the probability of packet loss in the network p as the probability that at least one node of the path will drop the packet after exceeding the maximum number of retries L at MAC layer, i.e.,

$$p = 1-\left(1-p_e^L\right)^k. \tag{16}$$

6.2 *Spectrum Sensing* State Analysis

In the *spectrum sensing* state, the TCP CRAHN adapts the *cwnd* size to prevent buffer overflow at intermediate node immediately preceding the sensing node. Let BF_{max} the maximum size of the buffer at each node. Then, if node i is performing sensing at time t, the residual buffer space at node $i-1$ can be approximated as

$$B_{i-1}^f = \left[B_{max}^f - \frac{W}{i-1}\right] DATA, \tag{17}$$

where W is the size of the *cwnd* at the sender node, i.e., $W = \frac{B_n}{DATA T_p}$. The congestion window W represents the number of transmitted but as yet unacknowledged segments, i.e., the segments that are in-flight. Moreover, the number of segments that are in-flight from the source to the node preceding the sensing node i is approximated by $\frac{W}{i-1}$. All these segments shall eventually arrive at node $i-1$ and take up the buffer space during the sensing duration, as node $i-1$ is unable to transmit to the downstream node i. The residual space, in terms of segments, left in the buffer of node $i-1$ (for the source to fill up additionally) is then calculated trivially as the difference between the maximum buffer capacity (B_{max}^f) and the in-flight segments $\frac{W}{i-1}$. To get the value of B_{i-1}^f in terms of bytes, we multiply the segment count with the packet size $DATA$.

1. This analysis can be extended easily for specific FEC schemes.

If sensing is performed by node 0 (the source) or node 1 (first next hop node), then $B_s = 0$ because the source node is forced to block its transmissions. Otherwise, the TCP CRAHN adapts the current rate so that node i does not overwhelm the buffer of node $i+1$. The average throughput during a sensing slot (B_s) can be derived using (17):

$$B_s = min\left\{\left(\sum_{i=2}^{k} B_i^f \frac{1}{kt^s}\right), B_n\right\} \tag{18}$$

$$\sim min\left\{\frac{1}{t^s}\left[(k-1)B_{max}^f - \frac{W}{k} ln(k-2)\right], B_n\right\}. \tag{19}$$

The probability to enter in the *spectrum sensing* state is the probability to have at least one node of the path starts sensing the current channel. Given t^s and T_p, the probability that at a given time interval a node is performing sensing is $p_s = \frac{t^s}{t^s+T_p}$.

Thus, the probability that at least one node of the chain is busy because of sensing activity can be derived as

$$P_s = 1-(1-p_s)^k. \tag{20}$$

6.3 *Spectrum Change* State Analysis

In the *Spectrum Change State*, the TCP CRAHN freezes the current state till the notification of spectrum handoff is received at the sender node. Thus, $B_c = 0$. The probability that a node detects the PU user during the sensing period (p_c) is the product of the independent probabilities of the PU being "on" ($\frac{\beta}{\alpha+\beta}$), the detection at the CR user being correct (p_d), and the CR user undertaking spectrum sensing at that time (p_s). Hence,

$$p_c = \frac{\beta}{\alpha+\beta} p_d p_s. \tag{21}$$

Thus, the probability to enter in the *spectrum change* state because at least one node can detect the PU activity during the "on" time is

$$P_c = \frac{\beta}{\alpha+\beta}\left[1-(1-p_d)^k\right]. \tag{22}$$

Thus, all the terms for calculating the TCP CRAHN maximum throughput B are available, which can now be calculated using (11).

6.4 Analytical Expression for TCP newReno

In the classical TCP throughput, we do not have the additional *spectrum change* and *spectrum sensing* states. Thus, the throughput in this case (B') can be approximated by

$$B' = \frac{DATA}{RTT}\sqrt{\frac{3}{2p}}.$$

Here, p and RTT are calculated in Section 6.1. However, if we there is no network feedback we can have additional packet loss for each link (i.e., p_e) given by the fact that when a node i transmits to node j, node j might be involved in sensing or switching activities, and thus be unreachable for

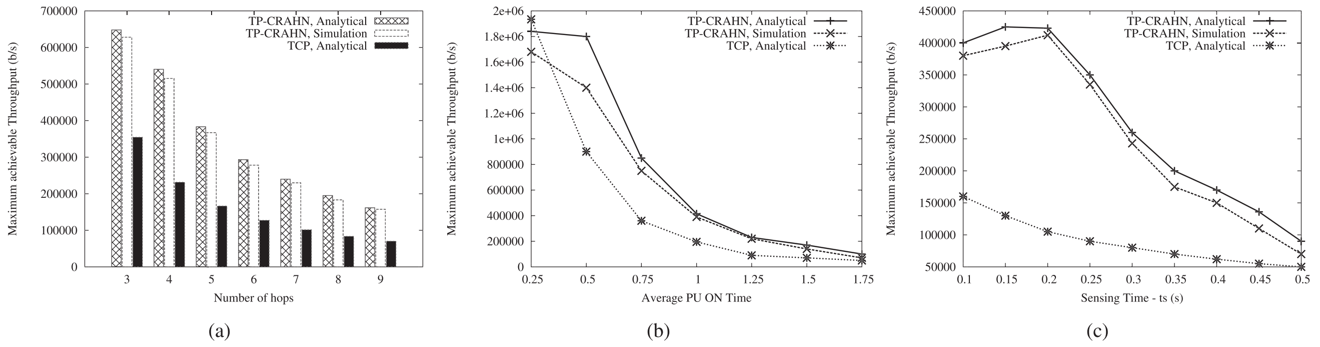


Fig. 6. The upper bound on the TCP CRAHN and TCP throughput is shown as a function of (a) varying path length, (b) PU "on" time, and (c) CR sensing duration.

data transmission. Again, we assume packet loss due to sensing or switching are independent from packet loss due to PU interference or channel error. Thus, the revised expression for p_e needed for calculating p is

$$p_e = p_I + p_T + p_s + p_c. \quad (23)$$

6.5 Validation of Analytical Models

In this section, we compare the analytical maximum achievable throughputs for TCP CRAHN (B) and TCPnewReno (B') under different network parameters. For this study, the nodes are stationary, and we assume Q-PSK modulation $M = 4, m' = 2, r_c = \frac{1}{3}$ [28]. Other parameters are $DATA = 1{,}500$ bytes and the transmission rate is 11 Mbps. Unless varied, default values of the number of hops (k) is assumed as 10, the sensing time (t^s) is 0.15 s and the transmission time (T_p) is 3 s.

As seen in Fig. 6a, the effect of the number of hops is significant for longer paths, as each additional node contributes to the end to end delay by spectrum sensing, and periodic spectrum switching, which are not predicted by classical TCP. As the PU "on" time increases in Fig. 6b, we find that the maximum throughput converges. This is because even though the spectrum change state is initiated by TCP CRAHN, the significant "on" time causes a complete stop in transmission by the source for large extent of time. Note that the fall in throughput is due to the intentional pause by the source (thereby protecting the PUs). This is different from the case for classical TCP, wherein the throughput loss is caused by segment losses due to collisions with the PUs. Finally, the fall in the maximum throughput is graceful for TCP CRAHN for different sensing durations owing to the intelligent flow control done by the source, as seen in Fig. 6c.

7 Performance Evaluation

In this section, we study the behavior of TCP CRAHN under the scenarios of 1) spectrum sensing, 2) spectrum change with PU activity, and 3) node mobility. To the best of our knowledge, there is no existing transport layer protocol that is designed considering the CR specific functions, and protocols devised for classical ad hoc networks cannot be compared fairly with our work. Rather, we use TCP newReno (henceforth referred to as TCP) as a benchmark and focus on how TCP CRAHN adjusts to each of the above CR scenarios through a stage-wise implementation of its modules. We have extended the NS-2 simulator for CRAHNs by modeling the activity of PUs and the multichannel operations performed by CR nodes [9]. At MAC layer, we have implemented the spectrum management functionalities including: spectrum sensing, decision, mobility and sharing functionalities. Periodically, each CR node senses the channel for a t^s time interval. In case of PU detection, it switches to another channel based on a round-robin channel selection algorithm. We consider a channel switching time of 5 ms. Otherwise, it may transmit on the current channel based on a CSMA/CA MAC scheme with acknowledgments (ACK) and frame retransmissions at the MAC layer. We also model the interference among CR nodes (secondary interference) and the interference between CR nodes and PUs (primary interference). In order to study the effect of the *cwnd* scaling, we consider five channels, $C = \{c_1, \ldots, c_5\}$, having varying raw channel bandwidth given by $\{B, \frac{B}{K}, B \times K, \frac{B}{K}, B \times K\}$, respectively, where $B = 2$ Mbps, $K = 2$. In addition, a priority queue is implemented at the link layer, as described in Section 4, and the allowed retries for the feedback packets in TCP CRAHN is raised to 20. Out of 100 nodes any source-destination pair is chosen forming a chain topology and we vary the number of flows in the path. A parallel chain is then created, which uses the same channel selection as our test chain topology. The corresponding nodes of the two chains are placed within transmission range of each other and this provides the link contention for the bandwidth calculation. The transmission ranges of the PU and the CR users are 300 m and 120 m, respectively, and the desired throughput $\tau_d =$ 800 Kbps. For maintaining the weighted average of the link bandwidths, we assign a weight of 0.2 to the latest sample and 0.8 to the previous recorded average.

7.1 Spectrum Sensing

The evaluation of TCP CRAHN during spectrum sensing is carried out in two parts—1) by observing the improvement in throughput resulting from the change in the *ewnd* (Section 5.3.1), and 2) the benefit of reduction of the sensing duration in the absence of PU activity (Section 5.3.2).

Figs. 7a and 7b show the end-to-end throughput for varying network loads as the sensing time of the nodes is increased to the maximum value $t^s = t^s_{max} = 0.3$, for a constant data transmission time T_p. In the first set of experiments, we disable the dynamic adjustment of the sensing time. We observe that TCP CRAHN outperforms

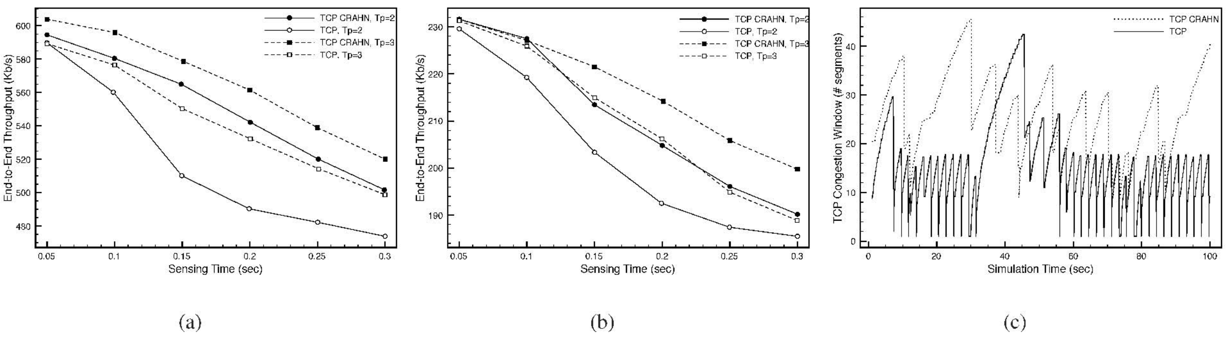


Fig. 7. (a) and (b) The effect of spectrum sensing on the throughput is shown for 1 and 5 flows. (c) Variation of the congestion window with time.

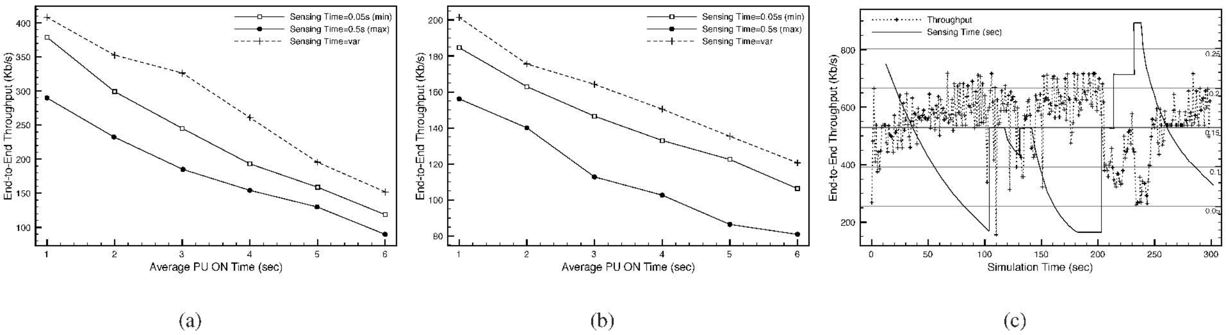


Fig. 8. (a) and (b) The effect of dynamically changing the sensing duration on throughput is shown for 1 and 5 flows. (c) A study of the throughput as a function of the varying sensing time.

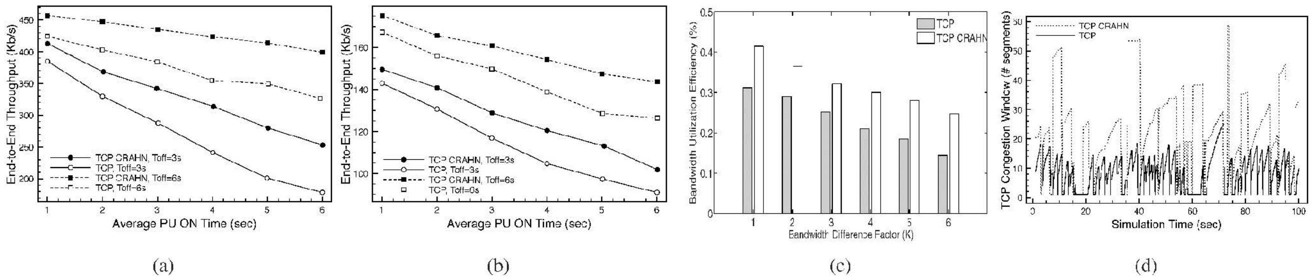


Fig. 9. (a) and (b) The effect of the bandwidth scaling adjustment on throughput is shown for 1 and 5 flows. (c) and (d) The bandwidth utilization efficiency and the *cwnd* scaling.

TCP significantly as it does not *stop* transmitting when the path gets disconnected, but transmits at a reduced rate to prevent a buffer overflow. This ensures a higher throughput in TCP CRAHN, as TCP suffers an RTO timeout whenever a node in the path undertakes sensing. This phenomenon is also seen in Fig. 7c, where the *cwnd* almost never reduces to 1 for the case $t^s = 0.15, T_p = 3$, unless there is congestion in the network (at sim. time = 70 s). Compared to this, TCP undergoes repeated timeouts during the sensing and the *cwnd* is forced to the slow start phase in the absence of true congestion. For the study of the dynamically changing sensing duration, we first define the PU operation as follows: We continuously vary the *on* time (T_{on}) of the PU on the x-axis, and measure the throughput for different PU *off* times, T_{off}, for 1 and 5 flows as shown in Figs. 8a and 8b, respectively. We first assign a a low PU susceptibility of 0.01 to a randomly chosen number of nodes S_{low} in the path, and for the remaining nodes, forming the set S_{high}, we set an initial high susceptibility value of 0.95. This ensures that TCP CRAHN changes the sensing duration from the maximum value $t^s_{max} = 0.28$ only for the nodes present in the set S_{low}. When the sensing time is varied, (shown by sensing $\mathrm{time} = \mathrm{var}$), we observe that the throughput improves significantly for 1 and 5 flows (Figs. 8a and 8b). The variation of the *cwnd* for throughput as a function of the changing sensing time is shown in Fig. 8c.The sensing time falls in a nonlinear manner (Section 5.3.2), and in the absence of PU activity, this improves the throughput. For successive PU missed detections, the sensing time is scaled to $\frac{1}{2} \times t^s_{max}$, then $\frac{3}{4} \times t^s_{max}$ and finally to the maximum value t^s_{max} to reduce the interference to the PUs.

7.2 Spectrum Change and PU Activity

From Section 5.4, we recall that when PU activity is detected, TCP CRAHN stops the source from transmitting and coordinates the use of a new channel. The source then modifies its *cwnd*, if the new channel on the affected link significantly changes the bottleneck bandwidth. We study the performance improvement in TCP CRAHN by considering the throughput, the bandwidth efficiency (ratio of the available bandwidth to average used bandwidth of the bottleneck link), and the variation in the *cwnd*.

A PU is placed on each of the five possible channels so that the channel (and hence, the bandwidth) change often and its effect is clearly demonstrated. Figs. 9a and 9b give the throughput for 1 and 5 flows, respectively, when PUs exist on the channel. We observe that the throughput improvement in TCP CRAHN increases with higher PU activity, formally defined as $\sigma = \frac{T_{on}}{T_{on}+Toff}$. For lower values of σ, i.e., when T_{on} is small, the channels are readily available and the gain in TCP CRAHN is due to the scaling of the *cwnd* alone. For higher values of σ, when T_{on} is large, it takes longer for the affected node to find a vacant channel. This delay dominates the network performance and by explicitly specifying the source to pause its transmission, TCP CRAHN prevents packet loss and improves the throughput.

The effect of *cwnd* scaling results in higher bandwidth efficiency in TCP CRAHN, as seen in Fig. 9c. Moreover, the performance improves when the difference in the raw bandwidths of the available channels (given by increasing the factor K) is higher, implying that the forced scaling of the *cwnd* is effective in fully utilizing the spectrum resource. The variation of the *cwnd* against time in Fig. 9d shows that TCP CRAHN responds to the changed bandwidth immediately.

8 Conclusions

The implementation of the TCP CRAHN protocol requires close coupling with the underlying link and network layers, especially during channel changes and during mobility-induced route outages. We note that complete implementation at the transport layer is only possible when link and network layer research on CRs have resulted in viable and standardized protocols, wherein the specific information available to the upper layers is known a priori. There are several assumptions made in the design of TCP CRAHN regarding this lower layer operation that may be different from the eventual practical implementations, and thereby require modifications in our approach when used on an actual CR testbed.

Acknowledgments

K. Chowdhury was supported by the US National Science Foundation (NSF) under Grant No. CNS-1265166. I.F. Akyildiz was supported by the NSF under Grant No. ECCS-0900930. Marco Di Felice was supported by Italian Miur PRIN-2006 project NADIR.

References

[1] I.F. Akyildiz, W.Y. Lee, and K. Chowdhury, "CRAHNs: Cognitive Radio Ad Hoc Networks," *Ad Hoc Networks J.*, vol. 7, no. 2, pp. 810-836, Elsevier, July 2009.

[2] A.O. Bicen and O.B. Akan, "Reliability and Congestion Control in Cognitive Radio Sensor Networks," *Ad Hoc Networks J.*, vol. 9, no. 7, pp. 1154-1164, Elsevier, Sept. 2011.

[3] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE J. Selected Areas in Comm.* vol. 13, no. 8, pp. 1465-1480, Sept. 1995.

[4] A. Capone, L. Fratta, and F. Martignon, "Bandwidth Estimation Schemes for TCP over Wireless Networks," *IEEE Trans. Mobile Computing*, vol. 3, no. 2, pp. 129-143, Apr.-June 2004.

[5] K. Chen, Y. Xue, and K. Nahrstedt, "On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks," *Proc. IEEE Int'l Conf. Comm. (ICC)*, pp. 1080-1084, May 2003.

[6] K.R. Chowdhury, M. Di Felice, and I.F. Akyildiz, "TP-CRAHN: A Transport Protocol for Cognitive Radio Ad Hoc Networks," *Proc. IEEE INFOCOM*, pp. 2482-2491, Apr. 2009.

[7] T. Dyer and R. Boppana, "A Comparision of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks," *Proc. ACM MobiHoc*, pp. 256-266, 2001.

[8] M. Di Felice, K.R. Chowdhury, and L. Bononi, "Modeling and Performance Evaluation of Transmission Control Protocol over Cognitive Radio Ad Hoc Networks," *Proc. 12th ACM Int'l Conf. Modeling, Analysis and Simulation of Wireless and Mobile (MSWIM '09)*, pp. 4-12, 2009.

[9] M. Di Felice, K. Chowdhury, W. Kim, A. Kassler, and L. Bononi, "End-to-End Protocols for Cognitive Radio Ad Hoc Networks: An Evaluation Study," *Performance Evaluation*, vol. 68, no. 9, pp. 859-875, 2011.

[10] S. Floyd and T. Henderson, *The NewReno Modification to TCP's Fast Recovery Algorithm*, IETF RFC 2582, Apr. 1999.

[11] G. Holland and N.H. Vaidya, "Analysis of TCP Performance over Mobile Ad Hoc Networks," *Proc. ACM MobiCom*, pp. 219-230, Aug. 1999.

[12] S.M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993.

[13] Y.R. Kondareddy and P. Agrawal, "Effect of Dynamic Spectrum Access on Transport Control Protocol Performance," *Proc. IEEE GlobeCom*, pp. 1-6, 2009.

[14] W.Y. Lee and I.F. Akyildiz, "Optimal Spectrum Sensing Framework for Cognitive Radio Networks," *IEEE Trans. Wireless Comm.*, vol. 7, no. 10, pp. 3845-3857, Oct. 2008.

[15] J. Liu and S. Singh, "ATCP: TCP for Mobile Ad Hoc Networks," *IEEE J. Selected Areas of Comm.*, vol. 19, no. 7, pp. 1300-1315, July 2001.

[16] C. Luo, F.R. Yu, H. Ji, and V. Leung, "Optimal Channel Access for TCP Performance Improvement in Cognitive Radio Networks," *Springer Wireless Networks*, vol. 1, no. 16, pp. 1-14, 2010.

[17] C. Luo, F.R. Yu, H. Ji, and V. Leung, "Cross-Layer Design for TCP Performance Improvement in Cognitive Radio Networks," *IEEE Trans. Vehicular Technology*, vol. 59, no. 5, pp. 2485-2495, June 2010.

[18] S. Mascolo, C. Casetti, M. Gerla, M.Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links," *Proc. ACM MobiCom*, 2001.

[19] T. Melodia, D. Pompili, and I.F. Akyildiz, "Handling Mobility in Wireless Sensor and Actor Networks," *IEEE Trans. Mobile Computing*, vol. 9, no. 2, pp. 160-173, Feb. 2010.

[20] M. Mathis, J. Mahadavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgment Options," IETF RFC 2018, Oct. 2006.

[21] N. Parvez, A. Mahanti, and C. Williamson, "An Analytic Throughput Model for TCP NewReno," *IEEE/ACM Trans. Networking*, vol. 18, no. 2, pp. 448-461, Apr. 2010.

[22] D. Sarkar and H. Narayan, "Transport Layer Protocols for Cognitive Networks," *Proc. IEEE INFOCOM*, pp. 1-6, Mar. 2010.

[23] A.M.R. Slingerland, P. Pawelczak, R.V. Prasad, A. Lo, and R. Hekmat, "Performance of Transport Control Protocol over Dynamic Spectrum Access Links," *Proc. Second IEEE Int'l Symp. New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, Apr. 2007.

[24] K. Sundaresan, V. Anantharaman, H-Y. Hsieh, and R. Sivakumar, "ATP: A Reliable Transport Protocol for Ad Hoc Networks," *IEEE Trans. Mobile Computing*, vol. 4, no. 6, pp. 588-603, Nov. 2005.

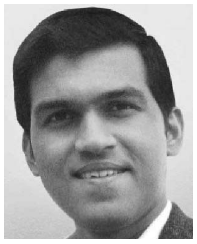
[25] F. Wang and Y. Zhang, "Improving TCP Performance over Mobile Ad-Hoc Networks with Out-of-Order Detection and Response," *Proc. ACM MobiHoc*, pp. 217-225, June 2002.

[26] H. Xiao, K.C. Chua, J.A. Malcolm, and Y. Zhang, "Theoretical Analysis of TCP Throughput in Adhoc Wireless Networks," *Proc. IEEE GlobeCom*, pp. 2714-2719, Nov./Dec. 2005.

[27] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP Fairness in Ad Hoc Wireless Networks Using Neighborhood Red," *Proc. ACM MobiCom*, pp. 16-28, 2003.

[28] D. Yuan, L. Zhang, and C. Gao, "Performance Analysis of RS-BCH Concatenated Codes in Rayleigh Fading Channel," *Proc. Asia-Pacific Conf. Comm.*, vol. 3, no. 1, pp. 315-325, 1999.

[29] X. Yu, "Improving TCP Performance over Mobile Ad Hoc Networks by Exploiting Cross-Layer Information Awareness," *Proc. ACM MobiCom*, pp. 231-244, Sept. 2004.



Kaushik R. Chowdhury received the BE degree in electronics engineering with distinction from VJTI, Mumbai University, India, in 2003, the MS degree in computer science from the University of Cincinnati, Ohio, in 2006, and the PhD degree from the Georgia Institute of Technology, Atlanta, in 2009. He is an assistant professor in the Electrical and Computer Engineering Department at Northeastern University, Boston, Massachusetts. His expertise and research interests include wireless cognitive radio ad hoc networks, energy harvesting, and multimedia communication over sensors networks. His MS thesis was given the outstanding thesis award jointly by the Electrical and Computer Engineering and Computer Science Departments at the University of Cincinnati. He won a best paper award at the IEEE ICC Conference in 2009 and again in 2012. He was also the recipient of a best paper award at the ICNC 2013 conference. He is a member of the IEEE.

Marco Di Felice received the laurea (summa cum laude) and PhD degrees in computer science from the University of Bologna, Italy, in 2004 and 2008, respectively. In 2007, he was a visiting researcher at the Broadband Wireless Networking Laboratory, Georgia Institute of Technology, Atlanta. In 2009, he was a visiting researcher at the Electrical and Computer Engineering Department, Northeastern University, Boston. He is now an assistant professor at the University of Bologna. His research interests include: modeling and simulation of wireless systems, including cognitive radio, mesh and vehicular networks, the distributed resources optimization, and the multi-hop communication in wireless networks. He authored more than 30 conference and journal publications on mobile and wireless network protocols, standards, and architectures.

Ian F. Akyildiz received the BS, MS, and PhD degrees in computer engineering from the University of Erlangen-Nürnberg, Germany, in 1978, 1981, and 1984, respectively. Currently, he is the Ken Byers chair professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, the director of the Broadband Wireless Networking Laboratory and the chair of the Telecommunications Group at Georgia Tech. In June 2008, he became an honorary professor with the School of Electrical Engineering at Universitat Politècnica de Catalunya (UPC) in Barcelona, Spain. He is also the director of the NaNoNetworking Center in Catalunya (N3Cat). He is the editor-in-chief of *Computer Networks (Elsevier) Journal*, and the founding editor-in-chief of the *Ad Hoc Networks (Elsevier) Journal*, the *Physical Communication (Elsevier) Journal*, and the *Nano Communication Networks (Elsevier) Journal*. He serves on the advisory boards of several research centers, companies, journals, conferences, and publication companies. His research interests include nanonetworks, cognitive radio networks, and wireless sensor networks. He received numerous awards from the IEEE and ACM. He is a fellow of the IEEE and ACM, in 1996 and 1997, respectively.